Aalto University
School of Electrical
Engineering

Master's programme in Smart Systems Integrated Solutions (SSIs)

Benchmark methodologies for the optimized physical synthesis of RISC-V microprocessors

Shashika Hansanie Samarasinghe

Master's Thesis
2023

Aalto University
School of Electrical
Engineering

Author Shashika Hansanie Samarasinghe

Title Benchmark methodologies for the optimized physical synthesis of RISC-V microprocessors

Degree programme Smart Systems Integrated Solutions (SSIs)

Major ELEC3064 - Smart Systems Integrated Solutions

Supervisor Assistant Prof. Ivan Vujaklija

Advisor Mr Ajay Ganesha (MSc)

Collaborative partner Nokia Networks and Solutions OY,Finland

Date 31 July 2023 **Number of pages** 60+2 **Language** English

Abstract

As technology continues to advance and chip sizes shrink, the complexity and design time required for integrated circuits have significantly increased. To address these challenges, Electronic Design Automation(EDA) tools have been introduced to streamline the design flow. These tools offer various methodologies and options to optimize power, performance, and chip area. However, selecting the most suitable methods from these options can be challenging, as they may lead to trade-offs among power, performance, and area. While architectural and Register Transfer Level(RTL) optimizations have been extensively studied in existing literature, the impact of optimization methods available in EDA tools on performance has not been thoroughly researched. This thesis aims to optimize a semiconductor processor through EDA tools within the physical synthesis domain to achieve increased performance while maintaining a balance between power efficiency and area utilization. By leveraging floorplanning tools and carefully selecting technology libraries and optimization options, the CV32E40P open-source processor is subjected to various floorplans to analyze their impact on chip performance. The employed techniques, including multibit components prefer option, multiplexer tree prefer option, identification and exclusion of problematic cells, and placement blockages, lead to significant improvements in cell density, congestion mitigation, and timing. The optimized synthesis results demonstrate a 71% enhancement in chip design performance without a substantial increase in area, showcasing the effectiveness of these techniques in improving large-scale integrated circuits' performance, efficiency, and manufacturability. By exploring and implementing the available options in EDA tools, this study demonstrates how the processor's performance can be significantly improved while maintaining a balanced and efficient chip design. The findings contribute valuable insights to the field of electronic design automation, offering guidance to designers in selecting suitable methodologies for optimizing processors and other integrated circuits.

Keywords Physical Synthesis, RISC-V microprocessors, EDA tools, ASIC implementation flow, SoC, floorplanning, performance improvement, PPA exploration

Preface

I am immensely pleased and honored to present this thesis, which marks the culmination of an arduous yet rewarding journey. As I reflect upon the countless hours of research, experimentation, and analysis that have gone into this work, I cannot help but acknowledge the invaluable support and guidance I have received from numerous individuals and institutions.

First and foremost, I would like to express my deepest gratitude to my supervisor, Assistant Professor Ivan Vujalkija. His unwavering commitment to academic excellence, relentless pursuit of knowledge, and insightful guidance have been instrumental in shaping the course of this research.I am truly fortunate to have had the opportunity to work under his mentorship.

I am equally indebted to my advisor, Mr. Ajay Ganesha, whose wisdom, expertise, and keen insights have significantly enriched my understanding of the subject matter. His constant motivation, meticulous attention to detail, and prompt assistance have been invaluable in navigating the challenges encountered throughout this research endeavor. I would also like to extend my heartfelt appreciation to my technical mentor, Nazeer Obi Abdulrahman and the entire Nokia Physical Design Team in Tampere, whose collaborative spirit, shared expertise, and constant encouragement have fostered an environment conducive to growth and innovation.I am grateful for the opportunity to have been part of such a talented and dedicated team.

Lastly, I would like to express my deepest gratitude to my family whose unwavering support, love, and understanding have been my pillars of strength throughout this challenging journey. Their patience, encouragement, and belief in my abilities have been the driving force behind my perseverance. Their presence in my life has provided me with the necessary inspiration and motivation to overcome obstacles and pursue my dreams.

To all those who have played a part in shaping this thesis, whether directly or indirectly, I extend my heartfelt appreciation. Without your support, guidance, and belief in my abilities, this work would not have been possible. It is with great pleasure and humility that I present this thesis, hoping that it contributes to the advancement of knowledge in the field and serves as a testament to the power of collaboration, mentorship, and unwavering support.

Otaniemi, 31 July 2023

Shashika H. Samarasinghe

Contents

Symbols and abbreviations

Symbols

α	Switching Activity Factor
μ_o	null bias mobility
C_L	Load Capacitance
C_{ox}	Gate Capacitance of the transistor
f_{clk}	Clock Frequency
I_{sub}	Sub-threshold current
V_{DD}	Supply voltage
n	Fabrication process dependent constant
P_{dyn}	Dynamic Power
V_{gs}	Gate source voltage
V_{th}	Threshold voltage
V_T	Thermal voltage
W/L	Aspect ratio of the transistor

Abbreviations

ALU	Arithmetic and Logic Unit
AOI	And-Or-Invert
ARM	Advanced RISC Machines
ASIC	Application Specific Integrated Circuits
AON	Always ON
CISC	Complex Instruction Set Computer
CMOS	Complementary Metal-Oxide-Semiconductor
CPU	Central Processing Unit
CSR	Control and Status Register
CTS	Clock Tree Synthesis
DEF	Detailed Placement and Floorplan
DFM	Design for Manufacturability
DSP	Digital Signal Processing
DVFS	Dynamic Voltage Frequency Scaling
EDA	Electronic Design Automation
ESD	Electro-Static Discharge
FPGA	Field Programmable Gate Array
FPU	Floating Point Unit
GDS	Graphic Data System
GUI	Graphical User Interface
HDL	Hardware Description Language
HLS	High Level Synthesis
HVT	High Threshold Voltage
HWLoops	Hardware Loops
IC	Integrated Circuit
IoT	Internet of Things
IP	Intellectual Property
ISA	Instruction Set Architecture
I/O	Input/Outputs
LEC	Logic Equivalence Check
LVT	Low Voltage Threshold
MBR	Multi-bit Register
MIPS	Microprocessor without Interlocked Pipelined Stages
MTCMOS	Multi Threshold CMOS
MUX	Multiplexers
PC	Program Counter
PPA	Power, Performance, Area
QoR	Quality of Results
RC	Resistance/Capacitance
RISC	Reduced Instruction Set Computer
RTL	Register Transfer Level

SBR	Single-bit Register
SoC	System-On-Chip
SP	Stack Pointer
STA	Static Time Analysis
SVT	Standard Voltage Threshold
TNS	Total Negative Slack
ULVT	Ultra Low Voltage Threshold
VHDL	Very High-Speed Integrated Circuit Hardware Description Language
VLSI	Very Large-Scale Integration
VT	Voltage Threshold
WNS	Worst Negative Slack
XPULP	Parallel Ultra Low Power extension

1 Introduction

The rapid advancement of technology has led to an exponential increase in the complexity of integrated circuit(IC) designs, and the physical synthesis stage of implementing an IC plays a critical role in achieving their desired performance, power, and area targets. In the context of physical synthesis of IC designs, there has been growing interest in exploring different benchmarked methodologies and options available in tools to optimize the design flow, improve the quality of results, and reduce the time-to-market to meet the high demand in the semiconductor industry[1]. In 2021, chips sold in the semiconductor industry surpassed 1 trillion units[2] and according to the results of European chips survey conducted by the European Commission, the industry anticipates that chip consumption will quadruple by 2030 compared to 2022 consumption[3]. To meet this huge demand and the rising performance expectations of chip consumers, semiconductor foundries are expected to continuously innovate and move towards lower transistor geometries following Moore's Law. However, due to the physical constraints of today's bulk chip materials, Moore's Law is no longer valid for chip sizes at the sub 10nm level. Nevertheless, advancements in 2D materials and heterogeneous systems enable further reduction in transistor size to achieve performance targets[1]. In addition to advancements in technology nodes and materials, the optimization of semiconductor devices through design-level improvements holds paramount significance.

Physical design refers to the process of transforming a circuit's logical description into a geometric layout that can be fabricated on a semiconductor wafer. With technological advancements and increased performance, the physical design process has also become surprisingly complex[4]. Improvements are continuously sought in performance, cost, flexibility, dependability, and maintainability, even within a single technology. Developers strive to complete each design stage more quickly to meet market needs. Implementing designs physically in lower node technology results in higher turnaround time and power consumption[5][6]. As the dimensions of transistors and interconnects shrink, designers must grapple with issues such as increased process variability, higher power dissipation, electromigration risks, signal integrity concerns, and stringent Design For Manufacturing(DFM) constraints. These challenges demand sophisticated solutions and meticulous attention to detail. Additionally, the intricate optimization requirements, such as power optimization, concurrent clock and data optimization, chip area reductions for cost benefits and device requirements, further add to the complexity[7]. To navigate these complexities successfully and achieve desired performance, designers must invest significant time and effort, employing advanced methodologies and tools to ensure reliable and efficient designs in lower technology nodes.

To aid this time-consuming and labor-intensive physical design process, computer-intensive design tools introduced by Electronic Design Automation(EDA) vendors are widely recognized by worldwide System-on-Chip(SoC) designers. Various options and methodologies have been developed to meet the performance, power, and area targets for these tools. However, simultaneously achieving these targets is challenging as they are contradictory in nature. Trade-offs are always required between routability,

time, and power consumption.Improving one aspect could negatively affect the others, and they must be tailored to each chip's needs[6]. Due to an inadequate emphasis on optimization methodologies, Application Specific Integrated Circuits(ASIC) typically suffer from a 10% reduction in optimization as measured in clock speeds, power, and area relative to full custom integrated circuits. Full custom IC design refers to the process of designing and implementing ICs from scratch, using individual components and layout techniques tailored specifically for a particular circuit application. ASICs are typically designed using highly automated logic, circuit, and physical synthesis EDA tools with available pre-designed and pre-verified cells and blocks[8]. Current research places significant emphasis on the need for future investigations in physical design optimization methodologies, which are commonly available in widely used EDA tools. While a considerable amount of studies have been conducted on architectural or Register Transfer Level(RTL) optimization techniques such as pipelining, power gating, and clock gating, there has been comparatively less focus on exploring the potential of physical design optimization methodologies. These methodologies play a crucial role in addressing power consumption and performance challenges. As examples, Tanya et al. have used clock gating in their research to present the optimization results compared to the design without clock gate[9], Paolo Mantovani et al. have used High Level Synthesis(HLS) tools to design pipe-lining effectively and optimized the processor[10], Chang et al. discussed about architecture and microarchitecture changes which can provide considerable amount of improvements in power dissipation[11] while Babu et al. have demonstrated algorithm level improvement for placement to improve performance and speed[12].

The evaluation of a chip's performance and the effectiveness of optimization methodologies often relies on the widely used benchmark criteria known as Power, Performance, and Area(PPA). Power measurement focuses on assessing the chip's energy consumption during operation, influencing factors like battery life and power management. Performance analysis considers the chip's operational speed, data processing rate, and computational efficiency. Area utilization evaluates the efficient use of silicon real estate on the chip, impacting its physical size, circuit density, and integration capabilities. By comparing PPA metrics, designers and researchers can gain valuable insights into the chip's overall quality, efficiency, and trade-offs, enabling them to make informed decisions on power management, performance optimization, and area utilization to develop more efficient and effective chip designs. These benchmarks are widely employed by EDA customers and researchers to evaluate the performance of ASIC designs[13][14][15].

This thesis aims to develop an optimized semiconductor device using optimization methodologies available in physical synthesis tools with floorplan exploration. The optimization methodologies and tool options in physical synthesis are expected to determine the optimal placement of standard cells and Intellectual Property(IP) block configurations, given the gate-level netlist of a design and associated files such as timing constraints and technology libraries. This optimization process aims to ensure that the post-place-and-route performance of a design aligns with Quality of Results(QoR) metric values, design rules and better performance when comparing PPA[16]. At the same time, it is expected to reduce the cost of resources like runtime or keep it

within a reasonable budget.For this implementation purpose, a widely used RISC-V processor is used. RISC-V is an open-source Instruction Set Architecture(ISA) based on the Reduced Instruction Set Computer(RISC) principles. It has gained significant attention in recent years due to its flexibility, modularity, and scalability, making it an attractive choice for a wide range of applications[17].

The scope of this thesis will be limited to analyze and compare optimized processor performance indicators after performing the physical synthesis including floor-planning steps where other physical design steps such as clock tree synthesis, routing will be not performed. To perform the synthesis, a cutting edge physical synthesis tool is used along with a latest lower technology libraries from a leading semiconductor foundry. The CV32E40P, a small and efficient RISC-V processor core design, which can be scalable using extensions available as open source is selected for this implementation process[18][19].For power analysis and timing analysis, QoR obtained from physical synthesis tools are used, even though there are specific tools for power and timing analysis for sign-off.

This thesis is organized in six main chapters as following outline and structure including this Introduction Chapter. Chapter 2 presents background information about the RISC-V processors and their architecture including their advantages, applications and limitations. Further, it provides the basis for the physical implementation and will briefly explain the ASIC implementation flow and the physical design flow, describing each stage involved and optimization methodologies. Chapter 3 describes the design steps followed in the thesis experiment by the user and methodologies performed to synthesize RISC-V CV32E40P microprocessor in EDA environment while Chapter 4 presents the performance results for synthesis iterations performed for different methodologies, different design specifications and different constraints. A detail discussion about the results obtained with explanations is presented in Chapter 5, while Chapter 6 presents the concluding remarks.

2 Background

The background chapter of this thesis delves into essential details regarding the processor architecture, structure, physical implementation procedure, and optimization methodologies employed to enhance IC performance. The selected processor core design, CV32E40P, is based on the RISC-V architecture and is known for its efficiency and scalability. It offers the flexibility to incorporate open-source extensions, enabling higher performance, code density, and energy efficiency[18]. The thesis explores the key features and functionalities of the CV32E40P processor core, providing an in-depth understanding of its internal structure, instruction set architecture, and key components such as the datapath, control unit, and memory subsystem.

Moving beyond the architecture, the physical implementation procedure is thoroughly discussed. The process encompasses various stages such as physical synthesis, floorplanning, placement, clock tree synthesis, and routing. Each step is explained in detail, highlighting their significance in achieving optimized performance and meeting the desired targets even though the thesis work is limited only to the physical synthesis and the floorplanning. Furthermore, the thesis delves into the specific optimization methodologies employed during physical implementation. These techniques may include power optimization approaches like clock gating and power gating, dynamic voltage and frequency scaling. The aim is to maximize the chip's performance, minimize power consumption, and optimize the overall area utilization to ensure efficient and reliable operation. By comprehensively exploring these aspects, the background chapter lays a strong foundation for the subsequent implementation and evaluation stages of the thesis.

2.1 RISC-V processors

Processor architectures are fundamental in shaping modern computer systems, mobile devices, embedded systems and other novel technological applications. Two main Instruction Set Architectures, known as RISC and Complex Instruction Set Computer(CISC), and RISC has gained popularity due to its faster and more efficient execution of instructions[20]. Over time, various ISA architectures, including x86, Advanced RISC Machines(ARM), and Microprocessor without Interlocked Pipelined Stages(MIPS), have powered a wide array of computing devices. x86, a widely used computer architecture, is based on CISC principles and finds extensive application in personal computers and servers. ARM, another significant architecture developed by ARM Holdings, follows the RISC approach and is commonly found in mobile devices, embedded systems, and low-power applications. MIPS, also a RISC-based architecture, is renowned for its simplicity and versatility, making it suitable for use in networking equipment, digital signal processors, and embedded systems across various industries[18]. However, these closed and proprietary architectures often stifled innovation and collaboration among researchers and industry professionals. In response to these limitations, the RISC-V architecture emerged as an open-source initiative at the University of California, Berkeley, in 2010. RISC-V sought to revolutionize processor design by promoting widespread academic and industrial collaboration, encouraging

innovation, and enabling easier customization and extensibility of processors[21].

The RISC-V architecture features a fixed instruction length of 32 bits, with several different instruction formats available. These formats include the R-type format for arithmetic and logical operations, I-type format for immediate instructions, S-type format for store instructions, U-type format for creating larger immediate values for operations as base formats and other sub formats for branch operations and other functionalities . These formats follow a consistent design principle of keeping the source registers(rs1 and rs2) and the destination register(rd) in the same position across all instruction formats as shown in Figure 1. This design choice aims to simplify the decoding process. However, there is an exception for Control and Status Register(CSR) instructions where 5-bit immediates (imm[x,y] bits) are used. In RISC-V, immediates are always sign-extended, meaning their sign bit is replicated to fill the remaining bits. To optimize hardware complexity, immediates are typically packed towards the leftmost available bits in the instruction, allowing for efficient decoding. Specifically, the sign bit for all immediates is consistently placed in bit 31 of the instruction, which facilitates faster sign-extension circuitry which improves the efficiency of the processor. The flexibility of these instruction formats allows for efficient encoding and decoding, while providing a wide range of instructions to meet various application requirements which is considered as a greatest advantage of RISC-V processors[17][19].

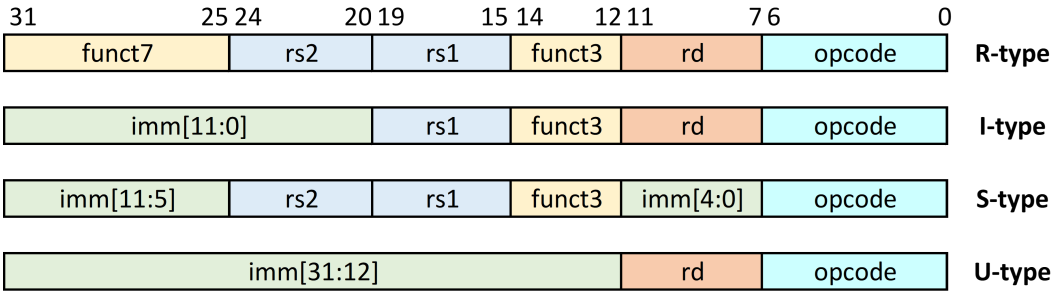


Figure 1: RISC-V core instruction formats: R-type, I-type, S-type and U-type, where rs bits define the register source, rd defines the destination, imm[x,y] defines the immediate bits and opcode for operation to be performed[19]

RISC-V utilizes a simple and consistent register file organization. It supports 32 general-purpose registers (X0 to X31), with X0 hardwired to zero. The register file can store both integer and floating-point values, and it provides efficient register access and manipulation operations. Additionally, RISC-V includes a separate program counter (PC) register and a stack pointer (SP) register for control flow and stack management[18][19]. The RISC-V architecture provides a comprehensive set of control flow instructions, including conditional branches, unconditional jumps, and procedure calls. These instructions enable efficient control flow manipulation, allowing for the implementation of loops, conditional statements, and function calls. The simplicity and regularity of the control flow instructions contribute to the overall efficiency and ease of programming with RISC-V [18][19]. RISC-V supports various addressing modes to facilitate memory access operations. It includes immediate addressing, base+offset addressing, and register indirect addressing modes. These

addressing modes enable efficient memory operations, such as loading and storing data, while minimizing the complexity of memory addressing calculations compared to other complex architectures[18].

The RISC-V ISA defines a rich set of instructions that cover a wide range of computational tasks. It includes instructions for integer arithmetic and logical operations, data movement instructions, control transfer instructions, memory access instructions, and privileged instructions for system-level operations. The extensibility of the RISC-V ISA allows for the addition of custom instructions and extensions, providing flexibility for domain-specific optimizations and accelerators[18]. The popularity and adoption of RISC-V have led to the development of a diverse ecosystem supporting the architecture. There are various implementations of RISC-V processors, ranging from small microcontrollers to high-performance multicore designs as emphasized in Chapter 1. These implementations include both open-source designs and proprietary solutions. Additionally, a wide range of development tools, compilers, simulators, and debuggers are available to facilitate RISC-V software development and system integration[22].

2.1.1 CV32E40P processor and its architecture

The CV32E40P processor has been chosen for the purpose of this thesis, and its key aspects are discussed to set the stage for the subsequent implementation and methodology sections. This includes an overview of the processor's architecture, its historical significance in the field, and a comprehensive exploration of its parameters, capabilities, and specifications. By providing this contextual information, the background chapter lays the groundwork for the subsequent sections, allowing for a deeper understanding of the design-specific details and the optimization methodologies employed in the thesis.

The CV32E40P is a RISC-V processor designed for embedded systems and Internet of Things(IoT) applications. It is an open-source processor based on the RISC-V ISA and offers a range of features suitable for low-power and resource-constrained environments[23]. The CV32E40P incorporates a 32-bit RISC-V core with a 4-stage pipeline and supports the RV32IMC instruction set, including integer, multiplication, and compressed instructions. It also includes hardware support for integer divide and square root operations, as well as a range of optional features like hardware performance counters, vectored interrupts, and debug capabilities. The CV32E40P processor is highly configurable, allowing system designers to tailor it to their specific requirements and optimize power, performance, and area trade-offs. This processor has gained attention in both academia and industry for its open-source nature, extensibility, and suitability for IoT and embedded systems applications[24]. It supports the floating-point extension and an optional custom extension called "Parallel Ultra Low Power extension(XPULP)". By incorporating the XPULP extension, the core achieves higher code density, improved performance, and reduced energy consumption. The architecture of the CV32E40P is optimized for Near-Threshold voltage operation, enhancing transistor efficiency[13]. Originally, the CV32E40P was derived from the OR10N Central Processing Unit(CPU) core, which was based on the OpenRISC

ISA. In 2016, it was renamed RI5CY under the PULP platform team and was made compatible with the RISC-V architecture. It was maintained as RI5CY until February 2020 when it was contributed by the OpenHW Group as CV32E40P[25]. The block diagram in Figure 2 illustrates the CV32E40P core with supporting extensions and its sub modules. A thorough comprehension of the high-level design of the processor's core, including its extensions and sub-modules, is crucial for analyzing timing and congestion issues. Understanding the interconnectivity between these modules enables the identification of the specific sub-module and interconnection responsible for potential problems. Equally significant is the ability to determine where the insertion of buffers or other elements might be necessary.

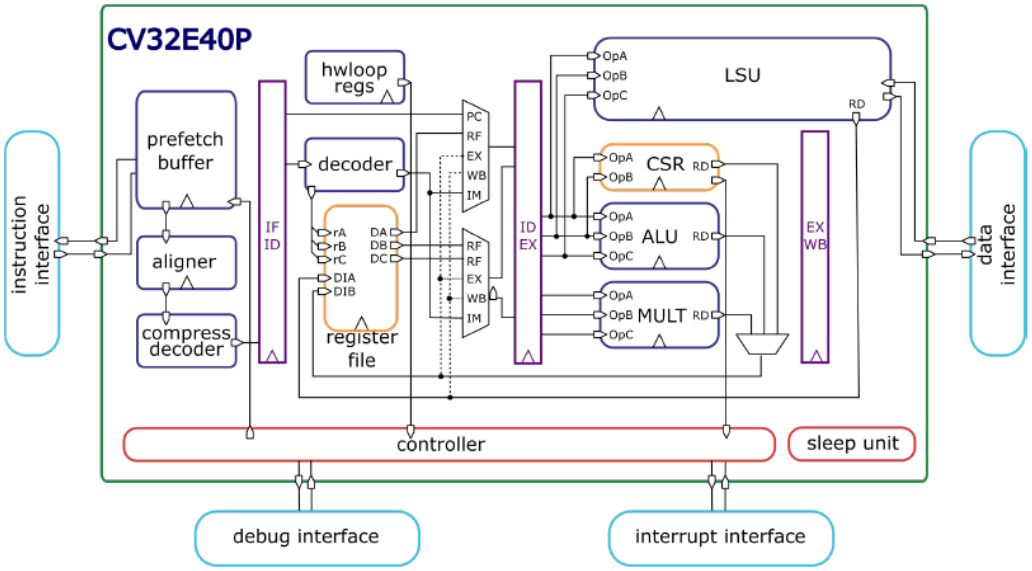


Figure 2: CV32E40P RISC-V Core, adopted from [25]

The CV32E40P core fully supports the Base Integer Instruction Set along with additional Standard Extensions such as M, C, Zicount, Zicsr, and Zifencei. The M extension is represented by the MULT block in the diagram, while the compress decoder block represents the C extension. It's important to note that the floating-point unit is optional in the CV32E40P core and is not depicted in the diagram. Additionally, the core incorporates PULP-specific extensions including Arithmetic and Logic Unit(ALU) extensions, hardware loops(HWLoops), post-incrementing load and stores, and multiply-accumulate extensions.

The CV32E40P processor incorporates two types of register files, namely latch based register files and flip-flop based register files, each serving specific purposes. The latch based register file is primarily recommended for ASIC implementation due to the less area consumption. Flip-flop based register files are recommended for Field Programmable Gate Array(FPGA) implementation by OpenHW group[25]. When the optional Floating-Point Unit(FPU) is instantiated in the CV32E40P processor, the register file is expanded to include an additional bank of 32 registers, labeled f0-f31, in addition to the general purpose registers unless the ZFINX parameter

is configured. These registers are stacked on top of the existing register file and can be accessed simultaneously. However, there is a limitation that a maximum of three operands can be read per cycle. To accommodate this, each operand address is extended with a register file select signal, generated by the instruction decoder when a Floating-Point instruction is decoded. This signal determines whether the operand is located in the integer register file or the floating-point register file. It's important to note that the forwarding paths and write-back logic are shared between integer and floating-point operations and are not duplicated. If the ZFINX parameter is set, there is no additional register bank, and the FPU instructions utilize the same register file as the integer instructions. This design ensures efficient utilization of resources and avoids unnecessary duplication of hardware components[13].

There are different parameters to be configured in the core design, based on the requirements. To enable the Floating point extension,FPU parameter of the top-level file cv32e40p_core.sv should be changed to 1. This will extend the register bank of the basic processor, but If the ZFINX parameter is set, there is no additional register bank, and the FPU instructions utilize the same register file as the integer instructions as explained in the above subsection. The optional support for HWLoops in CV32E40P helps short loops run more efficiently. Setting the PULP_XPULP option will allow them. Hardware loops enable numerous executions of a piece of code without the overhead of branching or counter updates. Hardware loops require no stall cycles to jump to the next step. CV32E40P implements performance counters according to the RISC-V Privileged Specification. Every counter has a 64-bit width. The parameter NUM_MHPMCOUNTERS, which has a range of 0 to 29 (with 1 as the default value), controls how many event counters are used[25]. The thesis extensively utilizes parameter configurations to enable various extensions and register files. Chapter 3 provides a detailed procedure for these configurations, explaining their specificities and implementation.

2.2 ASIC physical implementation flow

The advent of ICs in the modern computing era revolutionized the electronics industry. The concept of integrating electronic circuits onto a single semiconductor material was independently conceived by Jack Kilby and Robert Noyce in the late 1950s. In 1961, the first commercial semiconductor chip was born—an ingenious logical memory element known as a flip-flop, comprising four transistors and five resistors. This landmark achievement marked the beginning of a remarkable journey for semiconductor technology. Over the years, ICs have witnessed an incessant trend of miniaturization, driving a host of interconnected advancements. As devices shrink in size, they consume less power, operate at higher speeds, and have the capacity to accommodate an ever-growing number of functions due to their increased density[26]. The majority of ICs are ASICs, or Application-Specific Integrated Circuits. ASICs are designed with a specific application in mind, tailored to meet the unique requirements of a particular task or function. Unlike general-purpose ICs, ASICs are customized to optimize performance, power consumption, and cost-effectiveness for their intended use cases which is the ultimate objective of this thesis. This section gives a comprehensive guide to the step-by-step process involved in designing an ASIC. Also covered are various methods for enhancing design flow to get optimum performance numbers required for the specific design.

Application-Specific Integrated Circuit design flow is the process of creating custom-designed integrated circuits tailored to specific applications or functions. The ASIC design flow involves a series of steps and methodologies to transform a design concept into a physical chip. This process requires collaboration between designers, verification engineers, and manufacturing teams to ensure the successful realization of the ASIC design. The ASIC design flow typically begins with capturing the design specifications and requirements, which serve as the foundation for the subsequent design stages and goes until a successful tape-out with rigorous verification steps after each stage of design until fabrication.Figure 3 shows an overview of the ASIC implementation flow which will be discussed in the later of this section by elaborating each step. It begins by discussing the initial requirements gathering and specification phase, followed by architectural design, logic design, and verification stages. Subsequently, the chapter delves into the steps of physical design, including floorplanning and placement which is the main scope of this thesis. By providing a systematic overview of the ASIC design flow, this chapter equips the reader with a solid foundation for understanding the intricacies and nuances of ASIC development[26].

2.2.1 System specification and architectural design

Design specifications and requirements play a crucial role in the ASIC design flow as they serve as the guiding principles for the entire design process. These specifications define the functionality, performance, power consumption, and other essential characteristics that the ASIC must meet to fulfill its intended purpose. Design specifications provide a clear understanding of the desired behavior and features of the ASIC. They outline the input and output requirements, data rates, precision or accuracy

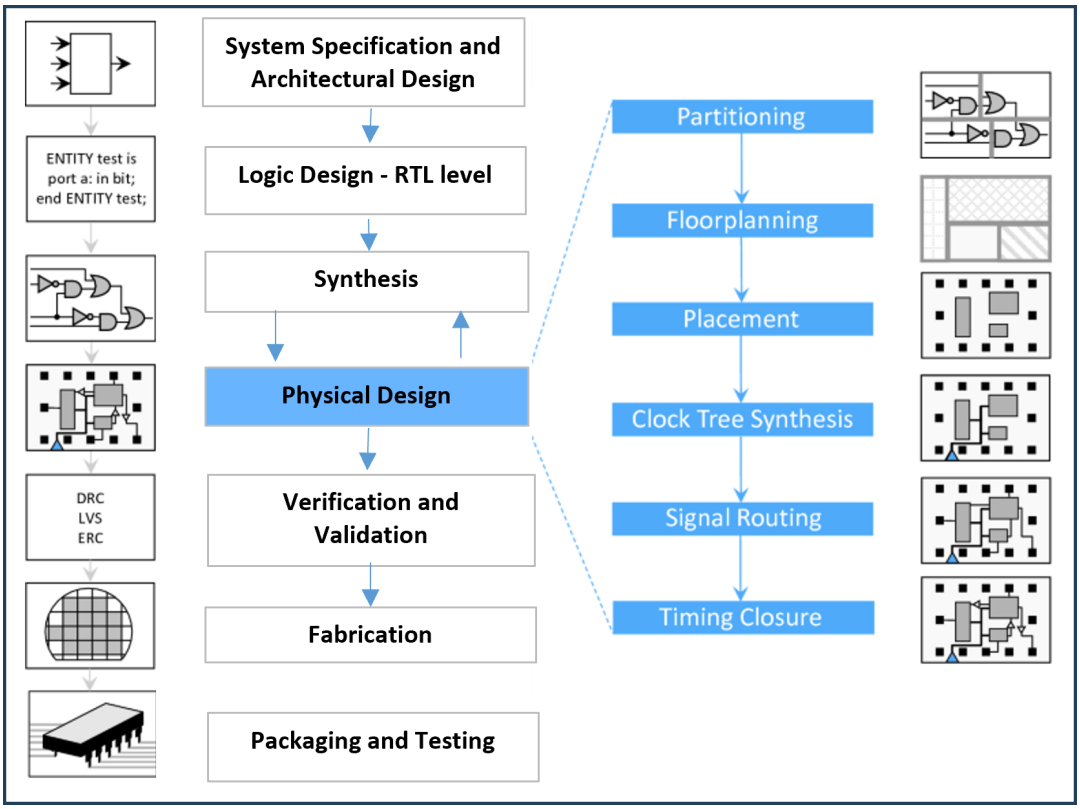


Figure 3: ASIC implementation flow[26]

levels, supported interfaces, and any specific algorithms or functions that the ASIC should implement[26]. In addition to functional requirements, design specifications also address performance aspects. They define the desired operating frequency or clock speed, response times, and any specific timing constraints that must be met. Performance requirements are critical for ensuring that the ASIC meets the desired speed and efficiency targets[17].

Power consumption is another significant consideration in ASIC design. The design specifications may include power budget constraints, specifying the maximum power consumption or the desired power efficiency. Power-related requirements help guide the design choices and optimization techniques to ensure that the ASIC operates within the power limits. Furthermore, design specifications may also cover physical constraints and limitations[17]. These constraints can include the available chip area, package dimensions, and thermal considerations. Physical constraints guide the chip's layout, floorplanning, and overall design decisions to meet the required size, form factor, and thermal requirements[26]. Design specifications are typically captured in a design specification document or a requirements document, which serves as a reference for all stakeholders involved in the ASIC design flow. Designers refer to these specifications throughout the design process to ensure that their design choices align with the desired requirements. The process of defining design specifications often involves close collaboration between the design team, system architects, product managers, and end-users or stakeholders. It is essential to have a clear understanding

of the intended application, user needs, and the business or technical goals of the ASIC to establish accurate and comprehensive design specifications.

The next step involves architectural exploration, where designers define the overall system architecture and identify the major functional blocks. This phase also includes making important decisions regarding the selection of technologies, such as the semiconductor process node and the type of ASIC (digital, analog, or mixed-signal), pipe-lining methods[27]. Overall, design specifications and architecture provide the foundation for the ASIC design process. They guide the design decisions, influence the selection of appropriate design methodologies and tools, and serve as benchmarks for verification and validation. By carefully defining and adhering to these specifications and following a suitable architecture, ASIC designers can develop customized chips that meet the specific needs of their target applications, ensuring the desired functionality, performance, and power characteristics.

2.2.2 Logic design

In the subsequent phase of ASIC design, the focus shifts towards implementing the defined specifications. Traditionally, this involved manually creating schematic drawings using components from a cell library. However, this manual approach was time-consuming and limited design reuse. To address these challenges,a language called Hardware Description Language (HDL) was developed. HDLs, such as Verilog and Very High-Speed Integrated Circuit Hardware Description Language (VHDL), enable designers to code the functionality of the design. Both languages serve the same purpose but have their own strengths and weaknesses[28].

Designers can represent the design at different levels of abstraction: behavioral, RTL, and structural. The behavioral level is a higher-level abstraction used to translate the architectural specification into code that can be simulated. Behavioral coding helps verify the authenticity and feasibility of the chosen design implementation. On the other hand, RTL coding describes the structural components and their connections, reflecting the functionality of the design. RTL coding is synthesizable and generates a structural netlist, which consists of components from a target library and their interconnections, similar to a schematic-based approach[29]. The design is typically coded in RTL style using Verilog, VHDL, or a combination of both. If needed, the design can be partitioned into smaller blocks to create a hierarchical structure, with a top-level block connecting all the lower-level blocks. This hierarchical approach aids in managing complex designs and promotes modularity and reusability[28].By utilizing HDLs and coding at the RTL level, ASIC designers can efficiently capture the design's functionality, simulate its behavior, and generate a structural representation in the form of a netlist. This shift from manual schematic drawings to HDL-based coding significantly enhances design productivity, flexibility, and design reuse capabilities in the ASIC design flow. Until this stage, the process is called as front-end designing[27].

2.2.3 Synthesis and Physical design

Logic synthesis is the process of converting a high-level hardware description of a digital circuit into a lower-level representation of basic logic gates and other pre-defined and pre-verified technology specific modules. The output will be a gate-level netlist which can be used in subsequent design steps[29]. Also at this stage, synthesis tools apply various algorithms and techniques to optimize the circuit's logic for area, power consumption, and performance. These optimizations involve simplifying logic expressions, identifying common sub-circuits, and reducing the number of gates and flip-flops, among other techniques which is the interest of this thesis including next coming physical design steps. Following the logic synthesis stage, the overall circuit design may include the integration of various other circuit types, such as analog circuits, digital circuits, and Electro-Static Discharge(ESD) protection circuits, which are designed at the transistor level if needed[26]. Alternatively, the gate-level netlist generated from logic synthesis can be directly utilized in the subsequent physical design steps. This includes creating a floorplan and performing placement of the chip's components. The results obtained from these physical design activities can be fed back into the synthesis process, enabling physical-aware synthesis. This iterative approach allows for optimizations that consider physical design constraints, leading to better overall results[27]. Those physical design activities, which involves floorplanning, placement, and routing of the chip's components will be discussed in the next section elaborately emphasizing more on the physical synthesis and floorplanning in Section 2.3.

2.2.4 Verification and Validation

Following the physical design stage, the design is extensively verified through various techniques such as static timing analysis, power analysis, and design rule checks to ensure that it meets the required performance, power, and manufacturability specifications[26].

Verification's primary goal is to ensure that the design is both functionally sound and producible. As a result, the verification's foundational criterion are the manufacturing restrictions. If the project plan specifies any hardware objectives, then various verification criteria can also be defined there. To make sure that the actual physical limits are never reached, the design restrictions are typically established with some additional dummy limits. The process of verification and validation begins early on, starting with the verification of the initial HDL code through tests and simulations. These tests can range from simple functional block tests to comprehensive system-level functionality tests. In some cases, such as with processor designs, simulations can even simulate the entire processor during program execution[17].

Another important step is the post-synthesis verification, where simulations are run using the gate-level netlist output. This allows for the evaluation of the digital flow results. It is essential to utilize simulators that can handle gate-level systems for accurate verification. Verification checks typically include formal, functional, timing, and physical verification[28]. Formal verification focuses on checking if the

design aligns with its specification, which can be a formal specification or property defined through mathematical methods. Model checking and equivalence checks are common techniques used in formal verification. Model checking verifies specific properties in the design model, while equivalence checks compare the synthesis results to the initial HDL structure, often referred to as Logic Equivalence Check (LEC)[17]. LEC is a critical step as it ensures the resulting chip will not be faulty. It typically involves three steps: setup, mapping, and compare. During setup, the initial RTL design and post-layout netlist are read, along with LEC settings such as instances to ignore. In the mapping step, both designs are flattened, and key elements like registers, inputs/outputs(I/Os), floating nets, and black boxes are mapped. The compare step compares the designs using the mapping established in the previous step[28].

Simulations are widely used for functional verification. Stimuli, in the form of inputs, are applied to the design, and the resulting outputs are compared to the expected outputs. Discrepancies in the outputs can reveal design errors or faults in the simulation, leading to improvements in both the simulation and the design itself. Simulating complex designs can be challenging, as the computation time increases with design complexity and simulation accuracy. It is important to acknowledge that verifying a complex design requires a large set of stimulus data combinations, and testing all possible combinations may be practically infeasible[17].

2.2.5 Fabrication, packaging and testing

Following the completion of functional and formal verification processes, the chip enters the fabrication stage, where it is sent to a fabrication facility(Fab house) for manufacturing.Fabrication involves the physical production of the chip according to the design specifications. It typically involves complex processes, such as photolithography, etching, and deposition, to create the integrated circuits on the semiconductor material. The fabricated chips are then packaged to provide protection and facilitate their integration into electronic systems[26]. After packaging, the chips undergo thorough testing and validation procedures to assess their performance, functionality, and reliability. Various tests are conducted, including functional testing, electrical testing, and environmental testing, to ensure that the fabricated chips meet the required specifications. Functional testing examines the chip's behavior and verifies its adherence to the intended functionality. Electrical testing assesses the chip's electrical characteristics, such as voltage levels and signal integrity, to ensure proper operation. Environmental testing subjects the chip to different conditions, such as temperature and humidity variations, to evaluate its performance under diverse operating conditions. The testing and validation phases are crucial to identify any manufacturing defects or inconsistencies that may affect the chip's functionality. Defective chips are typically rejected, while those passing the rigorous testing procedures are deemed acceptable for further use[4].

2.3 Physical Design flow

Physical design is a crucial stage in the ASIC design flow, where the logical netlist generated during logic synthesis is transformed into an integrated circuit layout. This layout is then converted into a Graphic Data System(GDS) file, which serves as the blueprint for IC manufacturing in the foundry[30]. The physical design process typically involves six key steps: partitioning, floorplanning, placement, clock tree synthesis(CTS), routing and Timing closure as per depicted in Figure 3 and will be discussed in detail in the next sections. These steps were performed using EDA tools, which are specifically designed for performing each stage in the design flow. The physical design results, which include the floorplan layout and the placement information of components on the chip, are incorporated back into the synthesis tool. By considering this physical information during the synthesis process, the tool can optimize the design more effectively, taking into account physical constraints, interconnect delays, and other layout-related factors. This results in a more precise and efficient design that aligns better with the specific target technology and manufacturing process.

2.3.1 Partitioning

Partitioning can be seen as a strategy that follows the "Divide and Conquer" concept, breaking down complex designs into simpler and more manageable blocks[26]. Designers may rapidly and effectively investigate various partitioning options because to the outline view's abstracted representations of logic modules. Only the content information required for an accurate representation of the module size and connectivity between modules is kept. Today's layout design tools provide facility for design engineers to explore the logic structure of even the largest designs in the Graphical User Interface(GUI) in real time because to this lightweight abstraction of the logic structures and they appeared with block utilizations for easy analysis. Designers can further concentrate on connecting timing-critical busses by filtering connectivity between blocks as modern tools have many user friendly options to view, select and filter real time on GUI based on the requirement [31]. Designer can decide on partitioning based on the connectivity between logical modules and their module size. One of the significant advantages of partitioning is the promotion of design reuse, which facilitates meeting timing constraints for individual blocks. Furthermore, partitioning enables the distribution and management of different design blocks among team members, facilitating collaborative work on the project.However, incorrect partitioning can result in inflexible boundaries that hinder optimization and degrade synthesis results. On the other hand, correct partitioning can greatly improve the synthesized outcome, leading to reduced compile time, simplified script management, and improved optimization possibilities[26][32]. According to Benini et al., memory partitioning is a highly effective solution for reducing energy consumption. Their research based on results collected on a set of embedded applications for the ARM processor, has shown average energy savings around 34%[33].

2.3.2 Floorplaning

In the floorplanning stage, the designer defines the area for the die, initial placement of various functional blocks on the chip, considering factors such as area utilization, performance goals, and signal integrity requirements. This step establishes a rough estimation of the block locations and their interconnections within a given set of boundaries or utilization factors.

Power planning is also included in the floorplaning, where the distribution of power and ground networks is determined. This involves designing supply rings, stripes, and grids to ensure proper power delivery to the circuitry while minimizing noise and voltage drop issues. It is essential to carefully layout the mesh grid to ensure that noise-sensitive areas are isolated and protected from noise interference. When placing generated memory blocks or intellectual property macros, several guidelines can help minimize total routing length[34]. Firstly, attention should be given to general placement strategies, such as block-driven or I/O-driven placement, which optimize the design by minimizing routing between blocks or keeping the distance between I/Os and blocks short. I/O port placement in the floorplanning stage of chip design involves strategically positioning the I/O cells between the core and chip boundary to enable communication with both internal and external components. Chip I/O placement determines the location of I/O pins and pads on the chip, while block I/O placement focuses on optimizing the placement of Block I/O pins within the core based on factors like timing, congestion, and chip utilization. Secondly, the orientation of macros becomes significant, as improper orientation can increase routing length when connecting macros. Lastly, aligning the macro's pins with the routing layer directions enables smoother routing without the need for additional vias near the start of the route[35]. Several optimization strategies have been employed by several researchers throughout last few years and have obtained improved results with proper floorplans[36]. Additionally, during the floorplanning phase, the placement of I/O pins and necessary circuits or specific areas, such as ESD protection, control logic or placement blockages, is also determined.

2.3.3 Placement

Placement is a critical phase where the exact positions of individual cells or components within the blocks are determined. The goal is to optimize the layout to minimize wirelength, reduce congestion, and satisfy timing constraints. Placement algorithms consider factors like cell density, interconnect proximity, and optimization objectives. To address the challenges posed by the complexity of placement in Very Large Scale Integration(VLSI) design, a multi-step approach can be employed. The placement process typically involves the following steps:

- Global placement: The goal of global placement is to generate an initial placement solution that provides a high-level view of the entire netlist, even if it violates some placement constraints. At this stage, modules or components are placed on the chip, and overlaps between them may exist.

- Legalization: After the global placement, the legalization step is performed to refine the initial placement solution and make it compliant with all placement constraints. It involves locally moving modules within the chip to eliminate overlaps and ensure a legal placement.
- Detailed placement: Detailed placement further refines the legalized placement by iteratively rearranging small groups of modules within local regions. The objective is to optimize specific metrics such as wirelength or routability. During this step, the positions of fixed modules are maintained while selectively rearranging others to achieve better overall placement quality.

The VLSI placement problem, extensively studied in the field of physical design, focuses on placing objects (modules or components) within a fixed die area. The primary objective is to avoid overlaps among objects while optimizing a cost metric, such as minimizing wirelength or enhancing routability[12]. Over the years, researchers have developed various algorithms and techniques to tackle this crucial step in the physical design of integrated circuits[12][37][38][39].

2.3.4 Clock Tree Synthesis

Clock Tree Synthesis is responsible for creating an efficient and robust clock distribution network throughout the chip The and it is one of the critical stages in the Physical Design as 70% of power is consumed by the clock[34][40]. It involves the insertion of clock buffers, clock tree optimization, and balancing to ensure clock signals reach all sequential elements with minimal skew and delay. The layout tool in the ASIC design flow is responsible for conducting the CTS immediately after cell placement and before routing. During CTS, the layout tool optimizes the placement and structure of the clock tree based on input from the designer. The designer typically specifies the number of levels in the clock tree and the types of buffers used at each level, taking into account the fanout of the clock signal[28].

In general, the number of levels in the clock tree is inversely proportional to the drive strength of the gates utilized in the clock tree. This means that if lower drive strength gates are employed, a greater number of levels will be required. Conversely, if higher drive strength gates are used, the number of levels in the clock tree can be reduced. The selection of drive strengths and the corresponding number of levels is a trade-off between signal integrity, power consumption, and timing considerations in order to achieve a robust and efficient clock distribution throughout the design[28]. The optimal scenario is to achieve a clock tree that is as symmetrical as possible. This means that the clock tree should have a balanced and uniform distribution of branches and levels. By achieving symmetry in the clock tree, the physical properties of the structure become more uniform and balanced reducing the clock skew [17]. Figure 4 shows two commonly used clock tree networks.

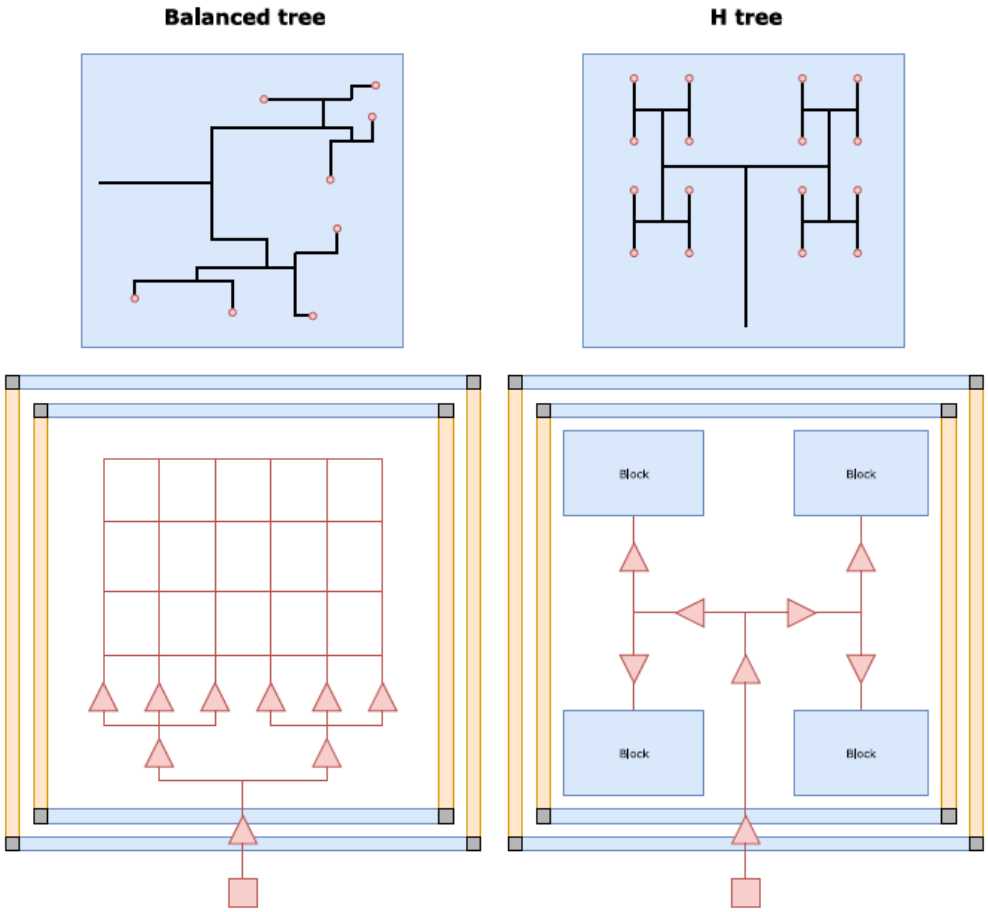


Figure 4: Clock tree networks, adopted from [17]. A balanced tree clock network is a clock distribution scheme where the clock signal originates from a single source and is then split into multiple branches, resembling a tree-like structure. Each branch further splits into more branches until it reaches the leaf nodes, which are the individual clock sinks.H-tree structure is another balanced clock network which resembles the letter "H," with a central source node and horizontal and vertical branches connecting to the clock sinks

2.3.5 Routing

The final step of the physical design is routing, where the interconnections between cells are established to establish the desired electrical connections. Routing algorithms consider factors like wirelength, congestion, and timing requirements while adhering to design rules and constraints.The routing phase of chip design consists of two main steps: global routing and detailed routing[28]. In global routing phase, the global router assigns a general pathway through the chip layout for each net. This is achieved by dividing the layout surface into several regions called grid cells and determining the shortest route through each region without actually laying down the physical wires.Grid

cells in the context of IC design refer to the discretized units that form a grid-like structure over the chip layout[17]. Once the global routing is completed, the detailed

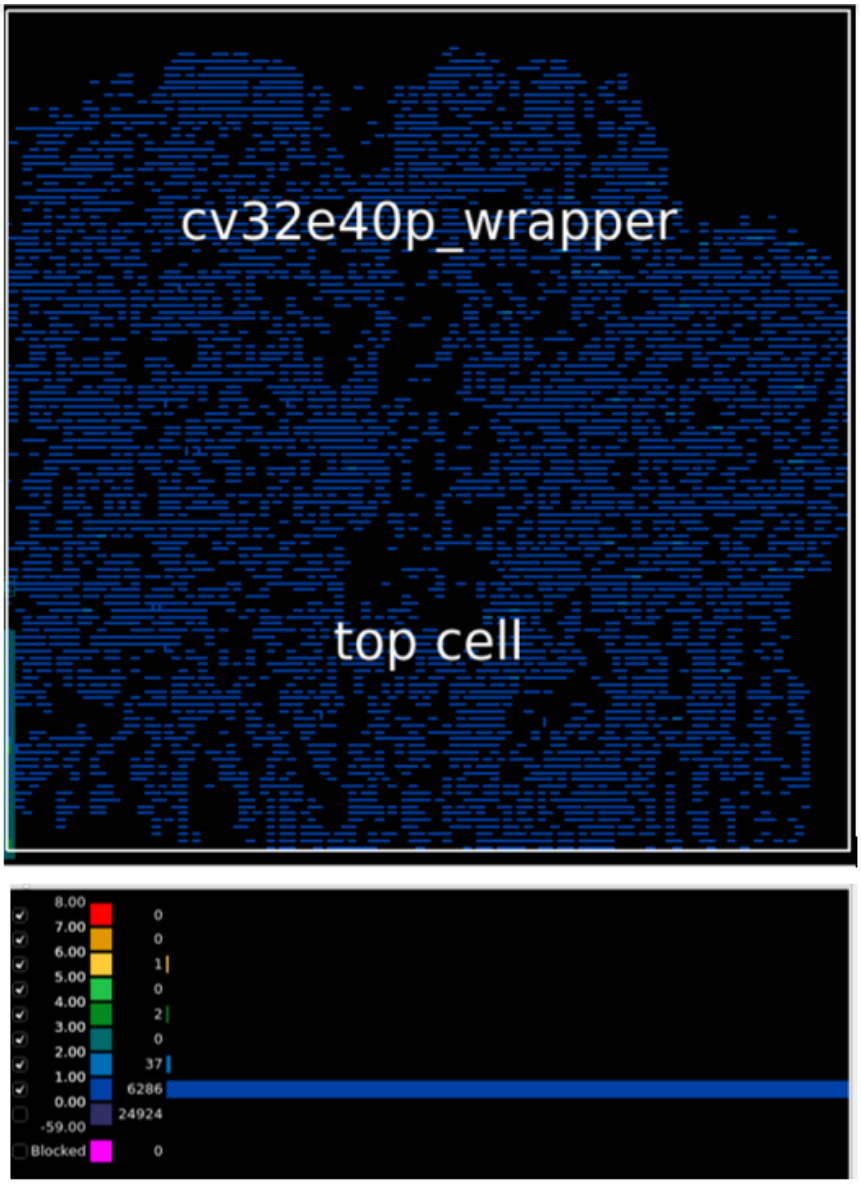


Figure 5: Global routing congestion map. The color scheme of routing map is essential in conveying information to the designer. The colors from blue to green, yellow, and red signify increasing levels of congestion in that order. Blue indicates congestion-free areas, while red indicates high congestion

routing phase begins. The detailed router utilizes the information obtained from the global route and proceeds to route the actual geometric wires within each region of the layout surface. This step involves the precise placement and connection of wires to establish the desired interconnections between different components of the chip. It is important to note that the runtime of the global route is an indicator of the placement quality. If the runtime of the global route is excessively long, exceeding the placement

runtime, it suggests that the placement may have resulted in congestion issues. In such cases, it is advisable to revisit the placement stage and focus on reducing congestion to achieve better routing results[17][28]. In synthesis tool after placement, global routing congestion map is available as shown in Figure 5. The global routing map is a visual representation generated by a synthesis tool that depicts the connectivity and availability of routing resources in a design. It provides a quantified scale, where higher numbers indicate greater congestion within the layout.When the global routing map displays red spots or pink spots, it indicates areas of congestion or blocked routes. Congestion occurs when there is a high demand for routing resources in a specific region, causing limited or unavailable paths for signal propagation. Blocked routes occur when there are physical obstacles or design constraints that prevent the proper routing of signals. Red spots in the global routing map are problematic because they signify potential timing and connectivity issues. Congestion can lead to longer signal delays, increased power consumption, and even signal integrity problems[28]. Overflows or blocked routes in these areas can hinder the successful completion of the routing process, requiring further optimization or redesign efforts to resolve the issues. According to this figure, the design is congestion is free and there are no blocked or red spots in the congestion map. Overall, the routing phase plays a crucial role in establishing the physical interconnections within the chip layout, ensuring proper signal propagation, and facilitating the functionality of the designed circuit[28].

2.3.6 Timing closure

Timing sign-off in ASIC physical design flow refers to the final stage where the timing analysis and optimization of the circuit are completed and verified. It is a critical step before the design is sent for manufacturing. Timing closure refers to achieving timing requirements such as setup time, hold time, and maximum operating frequency within the design constraints[17]. The design is subjected to detailed timing analysis to ensure that all paths meet the timing requirements. This includes identifying critical paths, analyzing slack values, and verifying that all setup and hold time constraints are satisfied. The goal is to minimize the negative slack and maximize the timing margin. Static timing analysis(STA) is performed to validate the timing performance of the design after optimization. It takes into account factors such as delays, capacitances, resistances, and clock uncertainties to ensure that the design meets timing specifications. Once STA is completed, timing sign-off is achieved. Timing sign-off is a crucial milestone in the ASIC physical design flow as it ensures that the design will operate correctly within the desired operating frequency and performance specifications[28].

2.4 Optimization methodologies

With the ever-increasing demand for faster and more power-efficient electronic systems, the goal of this research is to provide an optimized methodology to perform physical synthesis that effectively address the challenges associated with IC design. In the context of ASIC design, optimization methodologies are of paramount importance

to enhance performance, reduce power consumption, optimize area utilization, and meet strict timing requirements as explained in Chapter1. This section serves as a comprehensive exploration of the existing literature, presenting an array of optimization methods and the corresponding results reported. By examining the state-of-the-art in optimization methodologies, this chapter aims to provide valuable insights and establish a foundation for the subsequent research and analysis conducted in this thesis. This thesis work will specifically concentrate on one approach for each of floorplanning, timing optimization, congestion-driven optimization, and power optimization to compare there effectiveness in Chapter 3, although numerous other techniques from literature will also be explored and discussed in this section.

2.4.1 Power optimization

The demand for low power design is essential for minimizing power consumption in high-end systems. Achieving low power is closely linked to optimizing system performance. This is particularly important for high-performance applications such as Digital Signal Processing(DSP), microprocessors, and other similar systems. Power dissipation in Complementary Metal-Oxide-Semiconductor(CMOS) circuits can be categorized into three types: static power dissipation, dynamic power dissipation and short circuit power dissipation[9]. Static analysis considers the switching activity of inputs and calculates the power dissipation of a design based on the probability of switching propagation. The key difference between static power analysis and power analysis during logic synthesis lies in the inclusion of Resistance/Capacitance(RC) extraction from the layout in the former. On the other hand, dynamic power analysis is more precise as it utilizes a signal transition vector as input. This vector represents user-defined signal activity and provides a more accurate estimation of power consumption by considering actual input signals[30]. The dynamic power is given by:

$$P_{dyn} = \alpha f_{clk} C_L {V_{DD}}^2. \tag{1}$$

where α is the switching activity factor , f_{clk} is clock frequency, V_{DD} is supply voltage, and C_L is load capacitance. The clock frequency, and supply voltage are typically constant, as they are defined based on the specific design requirements of the application. To optimize power consumption in such scenarios, there are two possible approaches: reducing the load capacitance or reducing the switching activity factor[27].

The leakage power is a function of supply voltage and sub-threshold current and it is given by:

$$I_{sub} = \mu_0 C_{ox} \frac{W}{L} {V_T}^2 e^{1.8} e^{\frac{V_{gs}-V_{th}}{nV_T}}, \tag{2}$$

where μ_o is null bias mobility, C_{ox} is the gate capacitance, W/L is the aspect ratio of the transistor,V_T, V_{th}, and V_{gs} are thermal voltage, threshold voltage and gate source voltage respectively. n is a factor which depends on the fabrication method and process[41].

Based on the above equations,one option for power optimization is to decrease the load capacitance. The load capacitance represents the total capacitance that the

design's circuits need to drive, including the capacitance of interconnects, input/output pins, and other components. By minimizing the load capacitance, the power required to charge and discharge these capacitance can be reduced, leading to overall power savings. Another option is to reduce the activity factor. The activity factor refers to the proportion of circuitry that is actively switching or transitioning between logic states. By reducing the number of switching activities or optimizing the circuit design to minimize unnecessary transitions, the overall power consumption can be lowered. This can be achieved through techniques such as clock gating, power gating, and Dynamic Voltage and Frequency Scaling (DVFS)[27]. Both approaches, reducing load capacitance and minimizing the activity factor, aim to decrease the power consumption of the server while maintaining its required performance and functionality. The choice between these methods depends on the specific design constraints, performance requirements, and trade-offs that need to be considered for the server application.

In physical design, toggle rates refer to the frequency at which signals or nodes switch their logic states from 0 to 1 or vice versa. These toggle rates are crucial for estimating and optimizing the dynamic power consumption of a design. By identifying and targeting high toggle rate areas within the design, power optimization techniques can be applied to reduce dynamic power consumption. Here are a few methods commonly used and researched in last few years for SoC designs: clock gating, power shutoff, isolation cells, retention cells[9] which will be discussed in this section in detail. By leveraging the propagated toggle rates, the design can be optimized to minimize dynamic power consumption[15]. Annotating switching activity, and setting up the proper toggle rates can be performed within the physical synthesis tool. These methods target specific areas within the design where power reduction can be achieved without compromising functionality or performance.

Clock gating: Clock gating is a technique where the clock signal to certain circuit elements or registers is selectively enabled or disabled based on the activity of the associated logic[9][27][41]. By gating the clock signal, unnecessary switching activity and power consumption can be reduced. In clock gating, an enable signal is applied in conjunction with the clock signal of the circuit. When the enable signal is high or active, indicating that the circuit is ready to perform the required operation, the clock signal is allowed to propagate through the gated elements. In this state, the circuit functions as intended and carries out its operations. On the other hand, when the enable signal is low or inactive, the clock signal is effectively blocked from reaching the gated elements. This puts the circuit in a stand-by or idle mode, where power consumption is minimized because the inactive parts of the circuit do not undergo unnecessary switching activity. By preventing clock signals from reaching these idle elements, clock gating reduces dynamic power consumption. The enable signal acts as a control mechanism, determining when the circuit should be activated or deactivated based on the specific operational requirements. In a study conducted by Tanya et al., it was observed that clock gating led to a significant reduction in dynamic power by 85.5% and a reduction in leakage power by 40%, emphasizing the effectiveness of this technique in power optimization for integrated circuits[9]. This technique is commonly used in modern digital designs to optimize power consumption without sacrificing functionality or performance. This technique effectively reduces power

consumption by eliminating unnecessary switching activity in inactive parts of the circuit, ultimately leading to improved power efficiency and extended battery life in low-power applications. Figure 6 shows a simple clock gating circuit.

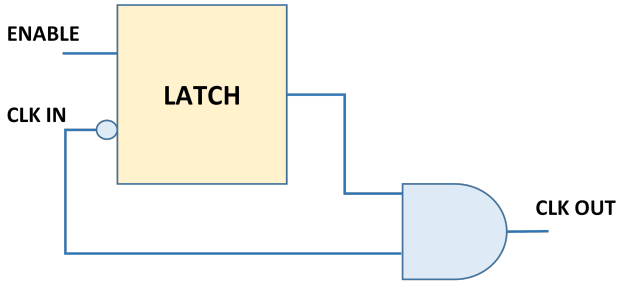


Figure 6: A simple clock gating circuit.The clock gate circuit controls the clock signal flow based on the enable signal and data input, allowing for power-saving opportunities by selectively enabling the clock when necessary for data processing operations

Switchable power domains: To achieve significant power reduction, power supply can be selectively switched off when a circuit is not in use by utilizing power switches. These power switches play a crucial role in implementing a leakage reduction technique called MTCMOS(Multi-Threshold CMOS). Power switches receive power from the main supply, which is referred to as the "Always ON"(AON) supply. The output of the power switches is switchable power that can be selectively supplied to different circuit components. Internally, these power switches incorporate multi-threshold cells, where High Voltage Threshold(HVT) cells are utilized when the circuit is in the off or standby mode. Conversely, very low threshold voltage cells are used in the operational mode. By employing high threshold voltage cells during standby mode, the leakage power, which increases exponentially with decreasing threshold voltage, is effectively limited. This is because higher threshold voltages reduce the leakage current through the cells, leading to reduced power consumption during periods of inactivity.When the circuit transitions into the operational mode, the power switches utilize very low threshold voltage cells to ensure optimal performance and functionality. These low threshold voltage cells enable efficient switching and minimize voltage drops, allowing the circuit to operate with improved power efficiency. The MTCMOS technique, facilitated by the integration of high threshold voltage cells in standby mode and low threshold voltage cells in operation mode, enables significant power reduction by effectively managing leakage power. By strategically controlling the power supply to different circuit components, power switches contribute to minimizing power consumption during idle periods and maximizing efficiency during active operation[9][41].

Voltage Threshold(VT) Cell Swapping: In addition, Low Voltage Threshold(LVT) cells, which offer high speed but also have the highest leakage current, can be replaced with HVT cells in power critical applications. This substitution helps mitigate leakage power and improve power efficiency. The choice of different VT cells is based on their characteristics related to leakage current and delay. LVT cells, despite their high-speed

performance, have significant leakage current. Standard Voltage Threshold(SVT) cells, on the other hand, exhibit moderate leakage and delay characteristics. Finally, HVT cells possess lower leakage but are associated with higher delays compared to the other types.To address power-critical paths within the design, existing cells along these paths are swapped with HVT cells. This substitution effectively reduces leakage power by utilizing cells with lower leakage characteristics. By selectively replacing LVT cells with HVT cells in power-critical areas, the overall power consumption of the design can be optimized[36][41]. A good analysis on different VT cells for their power consumption, area and timing performance has been done by Dinesh et al. considering only SVT,LVT and Ultra Low Voltage Threshold(ULVT) cells and their research shows the leakage power was at its lowest when only SVT cells were used in the design. When both SVT and LVT cells were incorporated, there was a significant increase of 31.52% in leakage power. Furthermore, when SVT, LVT, and ULVT cells were allowed, the increase in leakage power escalated to 96.01%. The results of their research are shown in a graphical form in Figure 7. To put it simply, using only SVT cells resulted in the least leakage power, but when LVT and ULVT cells were introduced, the leakage power substantially increased. However, this trade-off is accompanied by improved timing performance[42]. If the application is power critical, using HVT cells in the design can further reduce leakage power at the cost of timing. This approach strikes a balance between power efficiency and performance. While HVT cells may introduce some additional delay, the focus is on minimizing leakage power, especially in critical areas of the design. By strategically leveraging different VT cells, designers can achieve improved power optimization while maintaining acceptable performance levels[41].

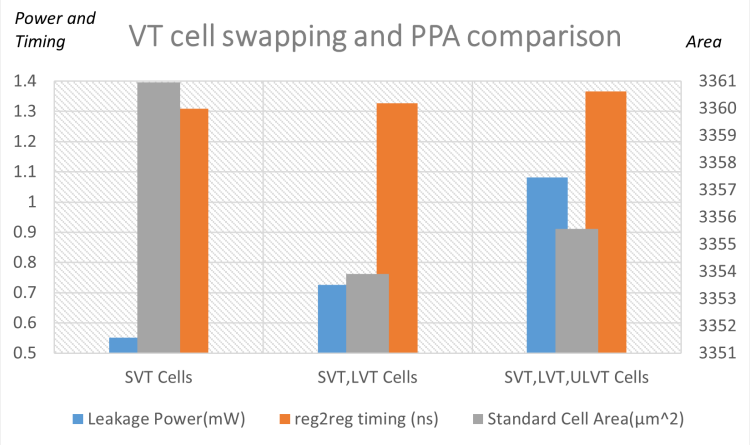


Figure 7: Comparative Analysis of Power Consumption, Timing performance and Area for Different VT Cells; data retrieved from [42]. As LVT and ULVT cells are introduced, performance improves, but this is accompanied by an increase in leakage current, impacting overall power consumption in the circuit

2.4.2 Timing driven optimization

Having a comprehensive understanding of timing reports and their definitions is crucial for conducting timing analysis and optimization. In order to meet the desired timing requirements, it is important to ensure that both the setup time and hold time exhibit positive slacks. Positive slack values indicate that the timing constraints are being met, allowing for proper functionality and reliable operation of the design. Slack refers to the amount of time by which a signal or path in a digital circuit is either early or late with respect to its required timing constraint. It is calculated by measuring the difference between the actual arrival time (or departure time) of a signal and its required arrival time (or required departure time) at a specific point in the circuit which is clearly interpreted in Figure 8. Setup time refers to the minimum amount of time that a data input signal must be stable before the active edge (usually rising edge) of a clock signal arrives. It ensures that the data signal has settled to its intended value before it is captured by the flip-flop or latch. Violating the setup time requirement can lead to data corruption or unpredictable behavior. Hold time, on the other hand, is the minimum amount of time that the data input signal must remain stable after the active edge of the clock signal. It ensures that the data remains stable long enough for the flip-flop or latch to capture and store it reliably. Violating the hold time requirement can result in metastability issues, leading to unpredictable outputs[43].

In the realm of ASIC physical design, delay optimization stands as a critical performance factor. Its significance is particularly pronounced in the development of microprocessors and real-time Application-Specific Integrated Circuits. Unlike area optimization, which involves aggregate metrics that can compensate for over and underestimates, delay optimization poses inherent difficulties. One of the key challenges in delay optimization lies in its path-based nature. Delay is highly sensitive to the individual components within the design, as it depends on the timing characteristics of the critical paths. Consequently, achieving accurate delay estimation and improvement necessitates a meticulous examination of each component's contribution along these critical paths. Furthermore, interconnect delay becomes a significant concern as the wire lengths increase relative to the transistor dimensions. Signal propagation delays along the interconnects can become a limiting factor for overall chip performance. Addressing interconnect delay requires careful routing, buffering, and the use of advanced signaling techniques, such as clock tree synthesis and global routing algorithms[44].

In high-performance circuits, a significant portion of timing optimization occurs during the placement stage. Traditional placement algorithms primarily focus on minimizing wirelength to achieve the timing objective. However, there exists a considerable gap between wirelength and actual delay, prompting the development of various methods to address this challenge. Recently proposed timing-driven placement methods can be broadly classified into two main categories: path-based and net-based methods[12]. Path-based methods directly aim to control the delays along critical paths. However, these methods can suffer from prohibitively high time complexity, particularly for modern large-scale circuits, as the number of paths grows exponentially. In contrast, net-based methods convert the timing constraint of each path into net

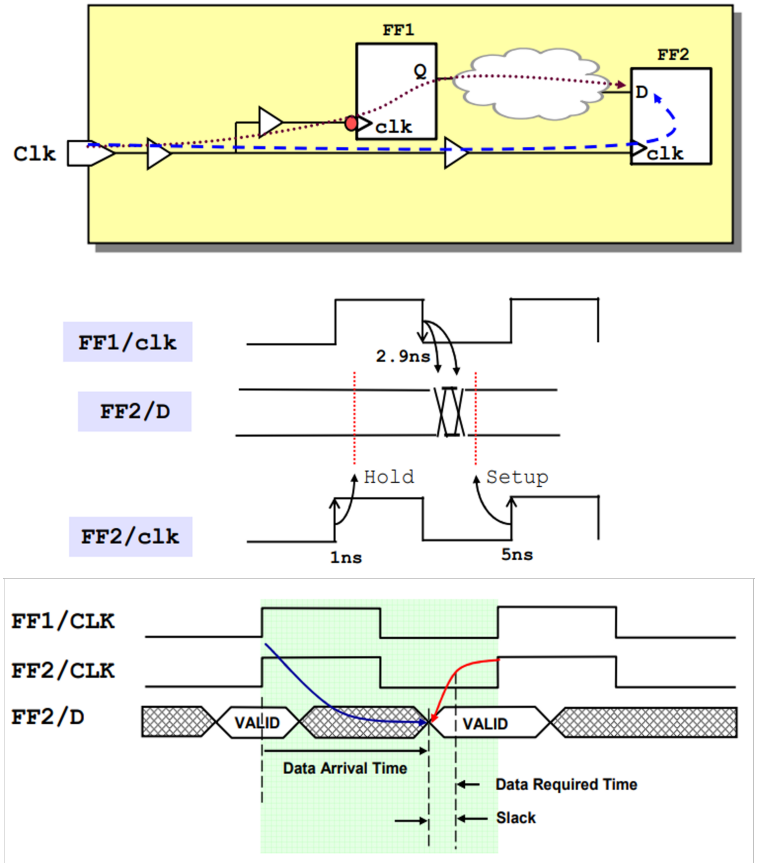


Figure 8: Timing definitions interpretation; adapted from [43]. Setup Time refers to the minimum time before the clock edge when the data must be stable to ensure proper data capture, while Hold Time denotes the minimum time after the clock edge when the data must remain stable. Slack represents the time difference between the actual arrival time and the required arrival time of the clock signal at the flip-flop

weights. These algorithms attempt to regulate the delay on a signal path by imposing separate constraints on individual nets within the path. However, this approach often leads to excessive constraints on the placement algorithm. This is because some nets in the path may already be considerably shorter than their bounds, implying that other nets could potentially accommodate additional delay. Extensive research has been conducted in the field of delay optimization, with scholars like Farrahi et al. and Sekhara et al. providing comprehensive literature reviews. These reviews have highlighted several net-based and path-based methods that were developed to address timing challenges. These methodologies were specifically designed to tackle issues related to the timing constraints of electronic designs[12][44].

Timing aware Multi Bit Register composition: In recent times, the incorporation of Multi-Bit Register(MBR) has become prevalent in commercial synthesis flows offered by EDA tools. This approach utilizes MBR instead of Single-Bit Register(SBR) as a means to reduce power consumption[45]. Traditionally, SBRs have been widely used in designs, where each register represents a single bit of information. However, MBRs

enable the grouping of multiple bits within a single register as shown in Figure 9. This grouping results in a reduction in the overall number of registers required in the design, leading to significant power savings. The process of MBR composition can be initiated early in the flow, specifically during logic synthesis, to achieve register power reduction. While early allocation of MBRs provides significant savings, it lacks crucial information related to placement and timing, which can impact the final outcome. Therefore, the majority of research on MBR composition focuses on identifying MBRs after global or detailed placement[45][46][47]. This approach seeks to strike a balance between reducing power consumption and managing potential trade-offs in timing slack, wire length, and routing congestion.

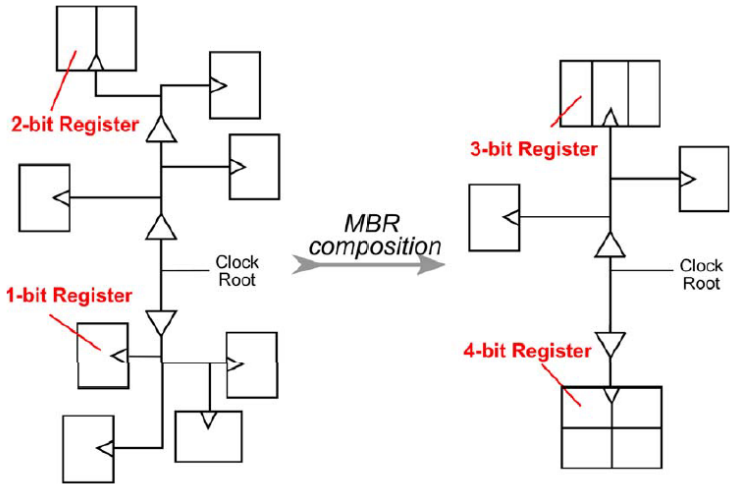


Figure 9: Multi Bit Register composition; adopted from [46]. In this design optimization, a 2-bit register and a 1-bit register have been merged to create a single 3-bit register, effectively reducing the number of individual registers. Similarly, four separate 1-bit registers have been combined to form a 4-bit register, streamlining the register count and enhancing overall circuit efficiency

2.4.3 Congestion driven optimization

Congestion-aware synthesis incorporates various optimization methodologies to address congestion issues during the placement phase. One such approach is the synthesis tool's ability to clump cells together based on a specified threshold. This enables the placer to group cells in close proximity, reducing the overall congestion in the design. By clustering cells together, the routing resources can be more efficiently utilized, minimizing wirelength and congestion-related problems. Additionally, congestion-aware synthesis incorporates the concept of "multiplexer(MUX) prefer" options. By setting this option in the synthesis tool, a congestion improvement strategy is employed specifically for designs that incorporate multiplexing logic. This strategy aims to alleviate congestion issues, even if it leads to increased runtime and potentially slightly larger area utilization. In the default flow, most multiplexers

are typically mapped to And-Or-Invert(AOI) logic to minimize the area footprint. However, in certain cases, this mapping approach can create congestion hotspots within the design. When mux prefer option is enabled, the strategy prioritizes the conversion of multiplexing logic that is likely to cause congestion into MUX trees whenever possible. MUX trees are a more congestion-friendly representation of multiplexing logic. By restructuring the multiplexing logic into MUX trees, the strategy aims to improve the congestion situation within the design[48]. To further enhance congestion reduction, a list of "don't use cells" can be provided. These are specific cells that are deemed problematic in terms of congestion. By excluding these cells from placement considerations, the placer can avoid potential congestion hotspots associated with those particular cells. Moreover, congestion-aware synthesis introduces the concept of inducing placement blockages. By strategically placing blockage regions in congested areas, the placer is guided to avoid allocating cells in those specific regions. This technique helps prevent worsening congestion and ensures that cells are placed in more favorable regions of the design. Collectively, these optimization methodologies in congestion-aware synthesis aim to minimize congestion-related issues during placement, improving routing resources utilization, reducing wirelength, and enhancing overall design quality[49].

3 Research material and methods

This thesis aims to develop an optimized and well balanced semiconductor device using various optimization methodologies available in synthesis EDA tools, with a specific focus on floorplanning. By evaluating and contrasting these methodologies, this research seeks to provide valuable insights into their strengths, weaknesses, and suitability for different design scenarios. The Research material and methods chapter will present a comprehensive overview of the experimental setup, data collection techniques, and evaluation metrics employed in this study, laying the foundation for an in-depth analysis of the optimization methodologies in subsequent chapters. The overview of the experimental setup with the procedural flow for this thesis objective is shown in Figure 10 where detailed explanation about the general ASIC physical design flow and the setup is given in Section 2.2.

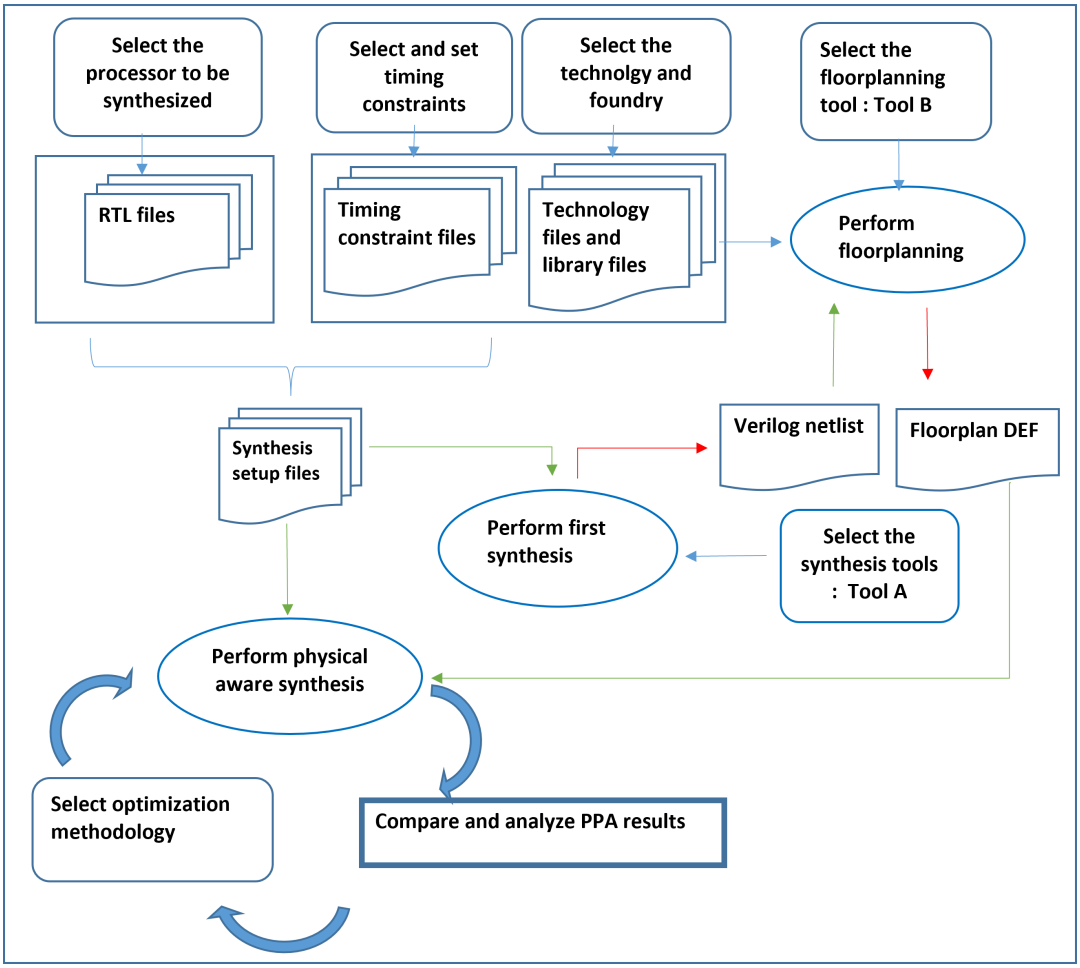


Figure 10: Thesis overview and experimental flow

3.1 Tool selection and setting up synthesis environment

Traditionally, the synthesis was performed based on wire load models. Synthesis findings based on wire load models were becoming increasingly unreliable and unpredictable due to shrinking semiconductor geometries. By merging synthesis and placement into a single common engine, EDA's advanced tools resolved this problem by introducing physically aware synthesis that have different optimization methodologies integrated and different options with great potential in future research[44][50]. To perform physical synthesis, the floorplanning step is essential to optimizing the placement of cells given constraints like core size, core utilization, pin placement, macro placement, and routing or placement blockages based on previous synthesis results as explained in Section 2.3.2. In the thesis project, Tool A, a popular physical synthesis tool, was used as the basic synthesis tool with its floor planning tool, Tool B to compare PPA and other performance related indicators such as run time between different processor designs with optimization techniques and timing constraints with an advanced lower technology library setup.

3.2 Performing first synthesis run experiments

Upon configuring the synthesis environment, a series of experimental synthesis runs were executed. Prior to commencing the synthesis process, meticulous analysis was performed on the design and configuration files. This involved a thorough examination of the RTL files, design specifications, and code documentation provided by the OpenHW Group and the Nokia physical design team. The purpose was to understand the functionality and design parameters of each RTL file before proceeding with physical implementation. The study also considered different options for the register unit of the processor, such as latch based and flip-flop based registers, to assess their impact on processor performance which were discussed in Chapter 2. Additionally, various constraint files were examined to evaluate the effects of changing input and output delays in different path groups during implementation. The synthesis environment was set up on a Linux server using licensed versions of Tool A and Tool B. Careful selection of appropriate tools, technology libraries, and design files was crucial. The RTL files, written in System Verilog and containing design specifications, and timing constraint files were uploaded to the workspace. The synthesis process was initiated without an accurate Detailed Placement and Floorplan (DEF) file. Instead, a predicted floorplan generated by the synthesis tool was used. While not entirely precise, this approach allowed the synthesis flow to progress and convert the RTL files into a gate-level netlist. This initial synthesis run served as a starting point to identify any potential issues or areas for improvement in subsequent stages of the design flow.

In order to evaluate the performance and functionality of the design core processor respective to the timing constraints, three distinct timing constraint files were employed, as illustrated in Table 1. Designers must make realistic specifications and restrictions, since unrealistic specifications lead to superfluous area , additional power, and/or timing degradation[26]. The implementation was based on a 5ns clock cycle, and variations in input/output delays, clock uncertainty, and output load were considered to

provide a comprehensive analysis. It is worth noting that the first synthesis experiments were exclusively conducted within the design environment of Tool A. As detailed in Section 2.1.1, the CV32E40P design incorporates various extensions that can be employed to enhance the performance of the design core processor. These extensions can be activated or deactivated by configuring the generic variables within the core module. Table 2 outlines the different parameter configurations used to enable specific extensions in the design, allowing for a thorough exploration of their impact on the overall design performance. Table 3 presents initial synthesis run experimental setups and these synthesis runs were instrumental in assessing the behaviour and efficiency of the design under various constraints and configurations. The obtained results serve as a foundation for further analysis and optimization in subsequent stages of the design flow. The comprehensive evaluation of the design core processor's performance, accomplished through the execution of multiple test cases and the exploration of various design extensions, showcases the versatility and potential of the CV32E40P architecture. By leveraging the capabilities of Tool A within the synthesis environment, valuable insights into the design's behaviour and opportunities for enhancement were gained.

Table 1: Timing constraint setups
Output Load: sets capacitive loading on output ports of the blocks, Input Delay: specifies the input arrival time of a signal in relation to the clock, Output Delay: defines the time it takes for the data to be available before the clock edge, Clock uncertainty: specifies the deviation of the actual arrival time of the clock edge with respect to the ideal arrival time

Timing constraints setup	**TS-1**	**TS-2**	**TS-3**
Clock period	5 ns	5 ns	5 ns
Input delays	60%	40%	10% - 80% based on the input port type
Output delays	60%	40%	25% -60% based on the output port type
Clock uncertainty	15%	15%	15%
Output load	1%	1%	1%

3.3 Performing physical aware synthesis

3.3.1 Floorplan creation

Physical aware synthesis is performed by giving physical constraints like the die size, pin placement and macro placement. Placement and floorplanning is a crucial step as it can improve the overall performance of the design. To perform the floorplanning and placement in this study, Tool B was employed to create multiple floorplans based on a predefined physical library setup and constraint file. The gate-level Verilog

Table 2: Different design setups

Design setup	Design description	Register File	Generic parameter configurations
DS-1	The basic processor core with a single performance counter	Flip-Flop Based	FPU=0, ZFINX=0, NUM_MHPMCOUNTERS=1, PULP_CLUSTER=0
DS-2	The basic processor core with a single performance counter	Latch Based	FPU=0, ZFINX=0, NUM_MHPMCOUNTERS=1, PULP_CLUSTER=0
DS-3	The processor core with a floating-point extension and a single performance counter	Flip-Flop Based	FPU=1, ZFINX=1, NUM_MHPMCOUNTERS=1, PULP_CLUSTER=0
DS-4	The floating point enabled processor core with a PULP (Parallel Ultra Low Power) extension and the maximum number of performance counters of 29	Flip-Flop Based	FPU=1, ZFINX=1, NUM_MHPMCOUNTERS=29, PULP_CLUSTER=1

Table 3: First synthesis experimental setups. Design specification setup from the Table 2 and timing constraint file from the Table 1 are combine together to perform a single first synthesis experiment

Test case	Design specifications	Timing constraints
FS – EXP 1	DS-1	TS-1
FS – EXP 2	DS-2	TS-1
FS – EXP 3	DS-1	TS-2
FS – EXP 4	DS-1	TS-3
FS – EXP 5	DS-3	TS-2
FS – EXP 6	DS-4	TS-2

netlist from the first synthesis run served as the foundation for these floorplans. The floorplan creation began by configuring the core utilization, core side ratios, and placing ports according to the design specifications using specific commands in Tool B. To incorporate floorplan information into the synthesis environment, the DEF file, which contained the floorplan details created by Tool B, was included along with the design's RTL files, tech libraries, and constraint file. This study focuses on five floorplan designs, considering only the rectangular shape, changing the side ratios and port placements with cell utilization of 50%. According to the design requirements, the floorplan can be selected and the optimum floorplan for a particular design is selected through a time-consuming iterative process and can be different from one

design to another. The detail setups for each design used in this research are illustrated in the Table 4. The area, power, timing, cell density, congestion, and run time of each iteration were carefully analyzed. Based on the results obtained, various optimization methodologies were applied to enhance the overall performance. These methodologies, collectively referred to as physical synthesis experiments, were employed to refine the floorplan and address specific areas of improvement. Through this comprehensive analysis and optimization process, this thesis aims to demonstrate the effectiveness of physical aware synthesis and the results obtained from these experiments will provide valuable insights for future design optimizations and contribute to the advancement of physical synthesis techniques.

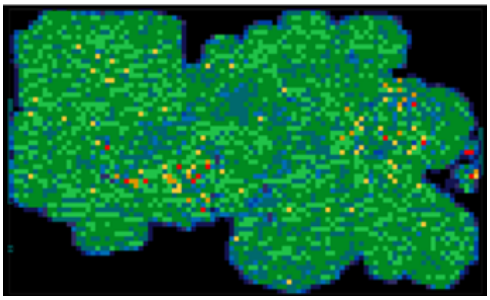
Table 4: Different floorplan setups

Floorplan setup	**Core shape**	**Port placement**	**Core utilization**
FP-1	Square	Inputs and outputs on one side with 1 pin spacing	50%
FP-2	3:5 side ratio	Inputs and outputs on one side with 1 pin spacing	50%
FP-3	3:5 side ratio	Inputs on left and outputs on right with 1 pin spacing	50%
FP-3	5:3 side ratio	Inputs and outputs on one side with 1 pin spacing	50%
FP-4	5:3 side ratio	Inputs on left and outputs on right with 1 pin spacing	50%

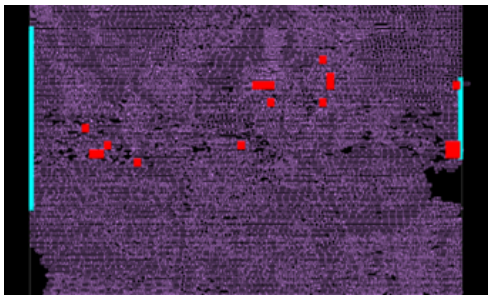
3.3.2 Applying optimization methodologies

After performing physical aware synthesis, next synthesis iterations were carried out to enhance timing, to reduce congestion, to reduce power based on the reported results. There are predefined options available in the tool that have not been extensively researched in the current body of literature, and they should be selected wisely as they have counter-effect on other performance metrics which is the primary objective of this thesis. In this study, multibit component preferred option with inferred default toggle rate[2.4.2], activating toggle rate to 12%[2.4.1], congestion aware synthesis which allows placer to clump cells together given a threshold[2.4.3], multiplexer prefer option[2.4.3], adding a list of cells as “don’t use cells”, inducing placement blockages in the floorplanning stage to identified hotspot areas are used as optimization methodologies. It is important to note that each of these methods were applied cumulatively. The first method involved enabling the multibit option without specifying a default toggle rate. The second method involved enabling a default toggle rate of 12% in addition to the multibit prefer option. Subsequently, congestion and cell density thresholds were defined. Finally, with all of these functionalities in place, the mux prefer option was enabled to further enhance cell density and timing. Additionally, cell defining and adding placement blockages were implemented based on the visual presentations of the synthesis results, aiding in achieving the desired outcomes. The optimization methodologies performed and how they have been effective in improving the performance are depicted in Chapter 4. Throughout this implementation process, clock gating and LVT cells from the foundry library for standard cells were used to enhance power and performance, as explained in detail in Section 2.4. However, the impact of utilizing these specific options was not investigated in this study, as they have already received significant attention in early research. Additionally, it should be noted that these options are not specific to the EDA tool and are therefore out of the scope of this thesis. For the implementation of the "don’t use cells" commands and placement blockages, the synthesis tool’s Graphical User Interface was utilized. The GUI provided a visual representation of the design and the synthesized netlist given the physical constraints which have been defined, allowing for the identification of congested areas, placement of cells and timing paths within them as shown in Figure 11.

The cell density map, as shown in Figure 11a, provides a visual representation of cell distribution and density across the chip’s layout. It offers valuable insights into chip area utilization by highlighting regions with high cell concentration (depicted in red) and those with relatively lower density (represented in blue). This map serves as a tool to identify potential congestion or imbalances in cell placement. In Figure 11b, the standard cell map is overlaid on the density map, allowing low-density areas to be filtered out, thereby focusing on highly dense spots. This aids in visualizing standard cells and hierarchical modules to pinpoint the causes of density-related issues. Figure 11c presents a zoomed-in view of reference cell names, enabling the identification of specific cell types that contribute to density and timing problems. Finally, Figure 11d displays a selected timing path with corresponding cell reference names, facilitating the identification of areas where timing violations occur and guiding optimization

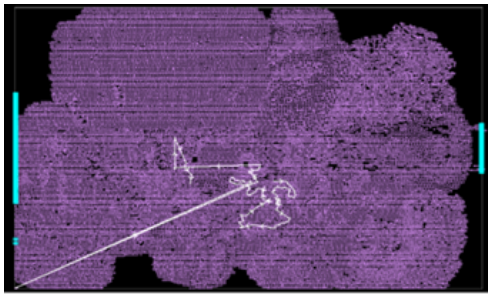


(a) Cell density map



(b) Cells placed in high-density areas

(c) Cell reference names



(d) Timing path visualization

Figure 11: Visual aids from synthesis tool GUI for optimization and troubleshooting after synthesis. The colored map from the GUI provides a visual representation of cell distribution across the chip core area. The GUI also enables zooming into high-density regions, allowing for individual cell identification and analysis for necessary improvements, while another visualization showcases critical timing paths

.

efforts to enhance overall performance. By utilizing the GUI options as shown in above provided figures, the cell congested areas, cells placed in those areas, and timing paths were identified, enabling the effective application of "don't use cells" commands and placement blockages for performance optimization.

4 Results

The evaluation of experimental results was conducted based on several key parameters, namely power, area, cell density, congestion, and runtime. The runtime refers to the duration taken for the entire synthesis process, including post-optimization steps, to complete. Power encompasses various components such as switching power, leakage power, and internal power, as indicated in the synthesis power report. Area represents the overall cell area, which includes combinational, non-combinational, and inserted buffer/inverter area. The timing results provide valuable information on the Worst Negative Slack(WNS), but in these reports, WNS refers to the worst slack where it can be positive as well, Total Negative Slack(TNS), and the number of violating paths. WNS refers to the largest negative timing difference between the required arrival time and the actual arrival time of a signal in a digital circuit, TNS refers to the cumulative negative timing difference across all paths in a digital circuit where Number of violating paths refers to the count of signal paths in a digital circuit that fail to meet the specified timing constraints. These timing definitions were further explained in Section 2.4.2 with greater detail. The cell density map visually illustrates the clustering pattern of cells within the design as discussed in detail in Section 3.3.2, while the congestion map offers insights into the design's routing capacity as described in Section 2.3.5. Each result is reported in the form of percentage as an increase or decrease with respective to a reference design. The results of the first experiment which is performed to select the register file to use in the design are illustrated in Table 5. Those experiments were detailed in the experiment section as FS-Exp 1 and FS-Exp 2 in Table 2 and Table 3. The experiment results with latch based file was taken as the reference and the performance of the flip-flop based register file respective to that reference design was depicted.

Table 5: Use of flip-flop based register vs Latch based registers

	Run Time	**Power**	**Area**	**WNS**	**TNS**
Flip-flop based	-28%	-33%	-2%	-82%	-94%

Figure 12 shows the cell density maps of the iterations using flip-flop based register file and the latch based register file respectively.

Then as explained in Table 1, timing constraints were modified and applied in different iterations. FS - EXP 1, FS - EXP 3, and FS - EXP 4 are the first synthesis runs without physical constraints provided for different timing constraints as tabulated in Table 3. FS-EXP 1 with TS-1 has been taken as the reference, and the corresponding performance variations with respect to that for TS-2 and TS-3 runs are shown in Table 6. The reference constraint file, TS-1, imposes tighter input and output delays, accounting for 60% of the clock cycle. In contrast, TS-2, which is the same constraint file, has relaxed input and output delays, amounting to 40% of the clock cycle. TS-3 includes varying input and output delays that are dependent on the type of input and output ports. These delays can range from 10% to 80% of the clock cycle, reflecting the different requirements for different port types.

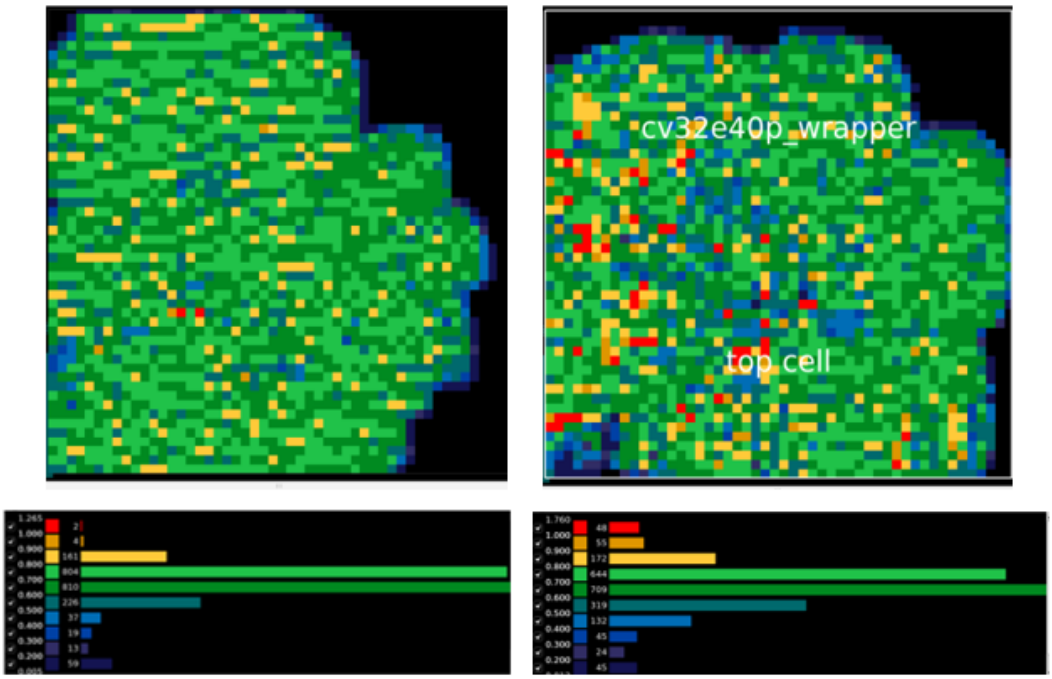


Figure 12: *left*: Flip-flop based register density map; *right*: Latch based register density map

Table 6: Performance comparison in timing constraint files

	Power	Area	WNS	TNS	No: of violating paths
TS -2	-27%	-30%	-424%	-100%	-100%
TS -3	-28%	-1%	-502%	-100%	-100%

Performance results variation of different design setup with different extensions are illustrated in the Table 7 compared to the basic processor unit. The corresponding cell density maps are presented in Figure 13. Following the synthesis phase, a fully synthesized high-level schematic design is accessible in the GUI. The synthesized design, depicted in Annex A, provides a comprehensive representation of the circuit at a higher level of abstraction of DS-2.

Table 7: Performance comparison in different processor design setups

	Cell count	Run time	Power	Area	WNS	TNS
DS-2	98.7%	148.0%	62.1%	76.2%	0.0%	0.0%
DS-3	220.5%	189.4%	146.5%	95.2%	100.0%	100.0%

Performance results obtained for different floorplan setups described in Table 4 are illustrated in Table 8. The corresponding cell density maps are attached to Annex B. The performance variations were compared with respective to the power, area, run

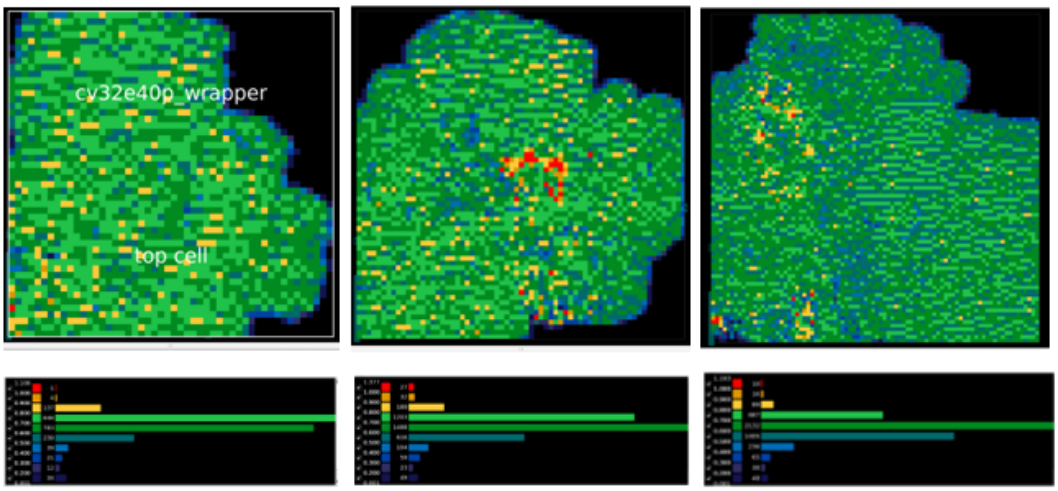


Figure 13: Density maps *left*: DS-1; *middle*: DS-2 *right*: DS-3

time and timing values obtained for the synthesis run without a floorplan.

Table 8: Run time, power and area comparison with different floorplan setups

Floorplan setup	Run time	Power	Area
FP-1	-43.2%	-2.7%	-0.2%
FP-2	-28.8%	-1.4%	-0.2%
FP-3	-47.7%	-0.8%	-0.3%
FP-4	-44.7%	-0.8%	-0.2%
FP-5	-50.8%	2.6%	-0.3%

Table 9: Timing comparison with different floorplan setups

Floorplan setup	Worst positive slacks in all path groups	Worst negative slacks in all path groups	Total negative slack	No.of violating paths
FP-1	575.0%	-25.0%	-24.3%	0.0%
FP-2	825.0%	-25.0%	-40.0%	3.3%
FP-3	650.0%	25.0%	13.0%	3.3%
FP-4	950.0%	0.0%	-7.0%	0.0%
FP-5	-50.0%	50.0%	39.1%	3.3%

Table 10 shows the area, power, timing and the run time variation of each optimization methodology enabled experiments compared to the basic design experiment which is not with those options enabled.

With these options, all the Reg-to-Reg, input and output paths were cleared for timing as they had positive slacks. Synthesis tool's GUI provides the opportunity to view slack details and timing paths in a graphical context. Figure 14 shows the positive slacks for all paths after optimization.

Table 10: Performance comparison in different optimization methodologies

Optimization methodology	Run Time	Power	Area	WNS	TNS
Multibit + inferred switching	-5.32%	542.09%	0.56%	-66.67%	-46.38%
Enabling toggle ratio of 12%	-18.09%	38.25%	0.59%	-33.33%	-15.94%
Congestion aware synthesis	-14.89%	40.83%	0.34%	0.00%	21.74%
Mux prefer option	-19.15%	49.31%	1.31%	-66.67%	-71.01%

Figure 14: Path slacks overview after optimization; all the paths have positive slacks

Upon applying each optimization methodology, the cell density was systematically compared, providing valuable insights into the efficiency of the design improvements and their impact on cell distribution across the chip core area. Figure 15 shows the cell density map without a floorplan and without any optimization methodologies. To reduce the cell density, as explained in the experimental section, the areas with high cell density were identified and improved using congestion aware synthesis options- 1, Mux prefer option- 2, adding troublesome cells as "Don't use" in placement- 3 and creating placement blockages- 4 respectively in order. The achieved cell density improvement throughout the process with each option is shown in the Figure 16. Figure 17 illustrates the congestion map after implementing all the experimented optimization methodologies where the congestion is less than 3 in all routing paths where the majority lies below 1 which is considered to be congestion free as explained in Section 2.3.5.

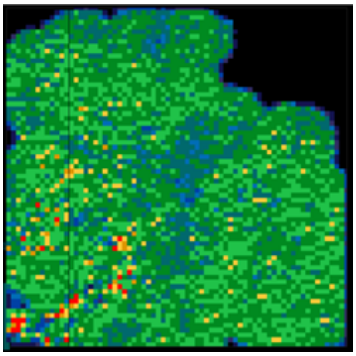


Figure 15: Cell density map without floorplan or optimization methodology applied

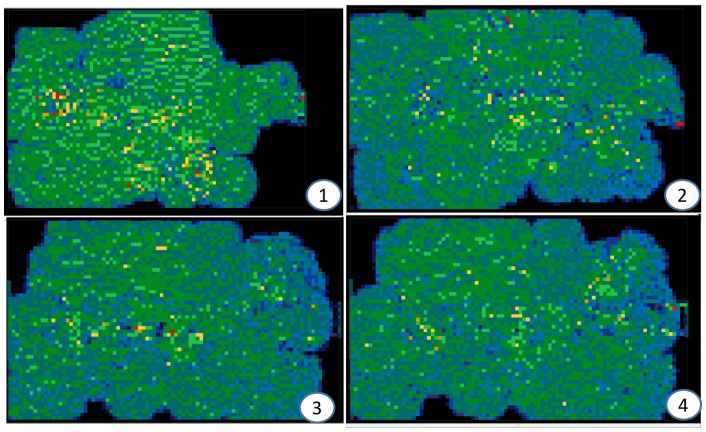


Figure 16: Cell density improvements from optimization methodologies. The color scheme progressively intensifies cell density, starting from blue to green, yellow, and red. A reduction in red spots and an increase in blue grid cells indicate improvements in cell density

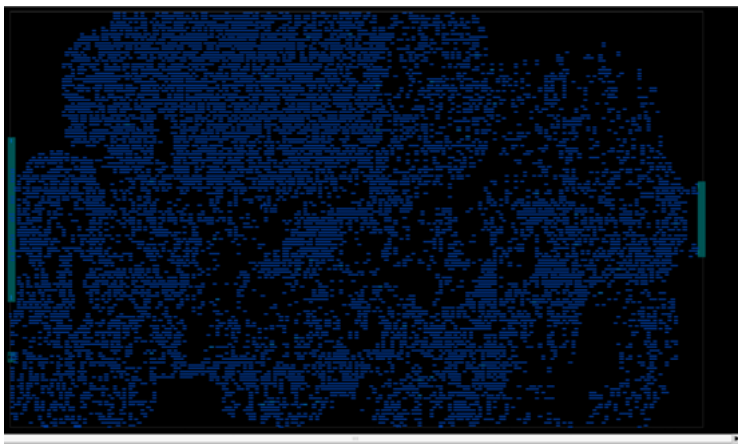


Figure 17: Congestion after applying optimization methodologies. The color scheme progressively intensifies congestion, starting from blue to green, yellow, and red. All blue signifies a congestion-free design which is the case in this design, while an increase in red indicates higher congestion levels

5 Discussion

The goal of this thesis was to develop an optimized semiconductor device using available optimization methodologies in physical synthesis tools, focusing on floorplan exploration. This thesis specifically analyzed and compared the performance indicators of an optimized open-source 32-bit RISC-V processor through the physical synthesis process, excluding clock tree synthesis and routing steps. The implementation utilized a cutting-edge physical synthesis tool and an advanced lower technology library setup from a leading semiconductor foundry. The selected processor core, CV32E40P, was a small and efficient RISC-V design that could be scaled using available open-source extensions. Power and timing analyses relied on QoR obtained from physical synthesis tools, although dedicated tools for sign-off analysis exist. Based on the results obtained in Chapter 4, it is clear that the steps and methodologies employed in this study have led to the successful development of a well-balanced and optimized semiconductor device, aligning with the initial objectives. In this chapter, each result will be discussed in detail, supported by relevant literature to provide a comprehensive analysis. The obtained results will be presented and examined to evaluate the effectiveness of the optimization process. By comparing the achieved outcomes with existing research, we can determine the impact and significance of the implemented methodologies. Furthermore, the results will be analyzed in the context of power consumption, performance metrics, and area utilization, considering the trade-offs and challenges faced during the physical design process. These insights will not only contribute to the current body of knowledge but also serve as a guide for future researchers and designers in their pursuit of efficient and optimized semiconductor devices.

In the first experiment, the goal was to determine the optimal register file for the specific design synthesized using the selected technological setup. The results demonstrate that the design utilizing the flip-flop register file exhibits superior performance in terms of timing, power, and area. The total negative slack for the flip-flop registered design improved by 94%, indicating a significant improvement in meeting timing requirements. While the area improvement was not substantial, with only a 2% advantage, it contradicts the recommendation by the OpenHW group's documentation for the CV32E40P processor, which suggests using a latch based register file for ASIC implementation based on the claim of better area utilization[25]. Hassan et al. concludes that they obtained potential benefits of using latches instead of flip-flops to enhance power efficiency, area utilization, and potentially even circuit speed. However this study obtained better results with flip-flops compared to using latches, despite the research indicating the potential benefits of using latches. Different factors such as timing constraints, processor design architecture and synthesis tool can influence the performance and effectiveness of different circuit elements used in a particular synthesis for its performance[51].

The results of this experiment to investigate the effect of timing constraints on the synthesis results as per Table 6 indicate that by relaxing timing constraints, it is possible to achieve improved performance in terms of meeting slack requirements. Furthermore, different input and output types, depending on their path and behavior, may exhibit varying input and output delays. Understanding the nature of these

different types and adapting the timing constraints accordingly can contribute to further enhancing the timing results. By considering and accommodating the specific characteristics of inputs and outputs, designers can optimize the timing performance of the circuit and potentially achieve even better results. Then the effect of the design size and the tool's ability to meet timing constraint and optimize the design were experimented. According to the processor manual, these enhance the functionalities of the processor, but it has higher cell counts compared to the basic setups leading to more power and area consumption respectively[25]. Based on the findings presented above, there is a clear correlation between the number of leaf cells in a design and its associated metrics such as area, power, and synthesis runtime. Specifically, it can be observed that as the number of leaf cells increases, there is a corresponding increase in the area, power consumption, and run-time of the synthesis iterations. This suggests that the complexity and size of the design, as indicated by the number of leaf cells, have a direct impact on these important performance parameters. With the advancements in synthesis tools which has the sole purpose of optimizing design and meeting given constraints, it is possible to meet timing requirements and generate positive slack even for larger designs with 98% cell increase. However, in the case of the third design which has more than 200% cell increase, the timing constraints could not be met.

The experiments conducted with varying floorplans demonstrated that the design's timing performance can be improved or compromised depending on the floorplan parameters. According to the article by Dhaval, The aspect ratio of a chip design plays a significant role in the availability of routing resources. When the aspect ratio is close to 1, it indicates a balanced proportion between the width and length of the chip. A close-to-1 aspect ratio in chip design provides ample routing resources in both horizontal and vertical directions, enabling greater flexibility for optimization during the placement process. If a chip design has more horizontal layers, it is preferable for the chip shape to be elongated, with a smaller width. This configuration allows for efficient utilization of the horizontal routing resources. Conversely, if there are more vertical layers in the design, it is advantageous for the chip shape to have a smaller length and a larger width. This arrangement optimizes the usage of the vertical routing resources[35]. As per the results obtained during this thesis, the square floorplan with the aspect ratio of 1 gave the best power optimization, while reducing total negative slack by 24% compared to the synthesis results with no floorplan. It didn't increase the total number of violating paths either. The floorplans with single-sided I/O ports also yielded the better timing results in terms of reducing negative slacks, while the floorplans with double-sided I/O ports exhibited the worse negative slacks compared to the reference. This observation can be attributed to the longer timing paths between inputs and outputs in the latter floorplan configuration, leading to increased timing delays and higher power consumption[36]. While the thesis has performed several iterations with different floorplans, the current manual approach requires extensive human intervention and is prone to inefficiencies. So, there is still a need to address the time-consuming and iterative nature of finding the optimum floorplan for a specific design. Future research should focus on developing automated and efficient techniques that streamline the floorplanning process, reducing the time and effort required.

It is essential to note that every design presents unique challenges, and the approach

to optimization should be tailored accordingly. The multibit register prefer method provided with this synthesis tool gave a notable initial improvement in timing but at the cost of power efficiency as the toggle rate was inferred and the tool couldn't decide the optimum toggle rate efficiently reduce the power. However, by refining the design by adjusting the toggle rate to 12% manually, a more balanced solution was achieved. The implementation of congestion aware synthesis, which involved introducing a flexible threshold for cell clustering, successfully mitigated congestion issues while preserving an acceptable level of cell density. This approach enabled cells to cluster in specific areas, while allowing for more relaxed placement in critical regions. This approach allowed for better utilization of the chip's core, resulting in improved timing and reduced congestion. Furthermore, the utilization of MUX trees instead of And-OR-Invert logic components proved to be an effective strategy in reducing cell density, congestion and timing as shown in Table 10. By favoring the inclusion of multiplexers in the design, the tool intelligently mapped multiplexing logic, resulting in a significant improvement in cell density and timing as described in the patent[48]. Finally including all of these options, a well-optimized design was achieved , demonstrating a 71% reduction in total negative slacks with improved other paths at a cost of 49% power increase and 1% area increase which can be concluded as a well balanced solution compared to the design with no optimization flows enabled.

Apart from timing and power analysis, the results obtained through various optimization technologies reveal significant improvements in cell density and congestion, providing valuable insights into the potential for overcoming the challenges posed by cell density and congestion in integrated circuit designs as depicted in Figure 16. The initial synthesis cell density map exhibited approximately 20 red spots, indicating areas of very high cell concentration, complemented by a mix of yellow and green spots representing high cell densities, indicating potential cell density issues. However, by incorporating a maximum cell clustering threshold of 60%, the number of red hotspot areas decreased by more than half. With the placer's maximum cell density set to 60%, despite the core utilization being limited to 50%, the placer has the flexibility to cluster cells up to 60% in certain regions. As a result, the cells will be placed within a range of 50% to 60% density, allowing for tighter packing of cells in those specific areas. This capability enables more efficient and optimized cell placement, contributing to better overall performance, reduced cell density issues and reduced routing congestion in the chip design. Subsequently, applying the Mux prefer option further reduced the red spots to only 3 areas and it was achieved by converting multiple AoI structures into more efficient mux trees, optimizing the cell distribution[48]. An in-depth analysis of the cells within those areas allowed for their inclusion in the don't use cell list, resulting in a reduction to 2 red hotspot areas. High drive strength cells and some AOI cells were identified as troublesome cells which cause the density issues. High drive cells typically require larger area and consume more power due to the additional transistors needed to provide the increased drive capability[49]. Eliminating or minimizing the usage of high drive strength cells during placement can indeed help reduce cell density issues, as demonstrated by the results. By incorporating the remaining hotspot areas as placement blockages during the floorplan creation stage, the number of red spots decreased to just 1,as blockages are used to restrict cell placements in specific defined

areas ultimately achieving a design that was both high cell density free and congestion free.

The results of this thesis has demonstrated the efficacy of diverse optimization techniques in enhancing the performance of semiconductor devices by addressing issues pertaining to cell density and congestion. The insights gained from this study enrich the existing body of knowledge in circuit design and synthesis, while also opening avenues for future advancements in the field.

6 Conclusion

In conclusion, the optimization techniques applied in this thesis have successfully addressed congestion and timing issues in the chip design. The experiments conducted with varying floorplans demonstrated that the design's timing performance can be improved or compromised depending on the floorplan parameters. Additionally, the identification and exclusion of specific problematic cells through the use of "don't use" attributes further enhanced the optimization process. By analyzing the high-density and congestion areas and selectively excluding cells with high drive strengths or specific characteristics, the tool was able to replace them with alternative cells, thereby improving density and alleviating congestion. The creation of placement blockages in the floorplanning stage provided another valuable constraint to guide the placement tool. By instructing the tool to avoid placing cells in the identified high-density areas, further reduction in cell density hotspots was achieved, leading to an overall improvement in the design's performance.

Overall, these optimization methodologies, including multibit register prefer option with toggle rate 12%, variable threshold clustering, Multiplexer(MUX) tree utilization, "don't use" cells, and placement blockages, collectively contributed to enhancing cell density, mitigating congestion, and improving timing in the chip design by 71% without significantly increasing the area. The results obtained demonstrate the effectiveness of these techniques in optimizing large-scale integrated circuits and highlight their potential for future design improvements in terms of performance, and efficiency. In this thesis, favoring the use of multiplexers for synthesis has been found to be the key factor contributing to its success. This discovery forms the main finding of this research. By emphasizing the importance of multiplexers in the synthesis process, this thesis highlights their effectiveness and their ability to enhance the overall outcomes of the synthesis process.

Looking ahead, there are a few promising avenues for future research and improvements in physical optimization that can further enhance chip performance and efficiency. While the thesis has made significant strides in congestion and timing optimization, there are still areas that warrant exploration to push the boundaries of chip design. One potential area of exploration is dynamic floorplanning, which involves dynamically adjusting floorplan parameters in real time based on design requirements and changing conditions using machine learning driven options[52]. This can be achieved through the integration of floorplanning tools and synthesis tools, enabling them to communicate and collaborate during the synthesis process. For instance, the initial synthesis results with a generic square floorplan and 50% utilization can trigger iterative floorplan adjustments by resizing and placing blocks within the main block.By intelligently adjusting the width and height of modules, efficient placement and routing can be achieved. Additionally, analyzing cell density and congestion maps can aid in blockage management. By dynamically configuring blockages within the floorplan, areas with high cell density or congestion can be identified and adjusted, alleviating potential bottlenecks and improving routing feasibility. Further for macro-placement, tool can employ data collected from previous results to perform macro cell placement automatically, generating an optimized and efficient initial placement that minimizes

congestion and timing issues. By predicting congestion, wirelength, and Total Negative Slack(TNS), the technology reduces the need for extensive manual tuning, streamlining the design process and ensuring a more optimal out-of-the-box placement that meets performance targets[53]. So as a future improvement, investigating strategies for automatically adjusting floorplan configurations to optimize performance and area utilization will have a good research and industrial impact using machine learning algorithms with clustering and modelling and heights of artificial intelligence.

This thesis mainly focused on performance improvement of the chip. As chip fabrication processes continue to shrink and become more complex, ensuring manufacturability also becomes increasingly critical. Future research should focus on developing physical synthesis techniques that take into account manufacturability constraints from the early stages of the design process. This includes optimizing for lithography, process variations, and yield improvement[54]. By incorporating Design For Manufacturability(DFM) considerations into the physical synthesis flow, designers can proactively address manufacturability challenges and enhance the overall yield and reliability of the chip.

These research directions, including the integration of dynamic floorplanning techniques with machine learning and artificial intelligence, and design for manufacturability considerations, hold great promise for advancing physical synthesis methodologies beyond the current state-of-the-art. By addressing these areas, chip designers can further optimize chip performance, improve power efficiency, and ensure robustness in the face of manufacturing constraints. In summary, this thesis demonstrates the effectiveness of optimization methodologies in improving large-scale integrated circuits. The achieved results highlight the potential for future advancements in terms of performance, efficiency, and manufacturability. By continuing to explore these research directions and addressing emerging challenges, the field of physical optimization will continue to evolve, driving innovation in integrated circuit design and further pushing the boundaries of chip performance and optimization.

References

[1] S. Wang, X. Liu, and P. Zhou, "The Road for Two-Dimensional Semiconductors in Silicon Age," *Advanced Materials*, vol. 34, no. 48, Feb. 2022, doi:10.1002/adma.202106886.

[2] J. Bish DS, K. Ramachandran, and P. Lee, "A new dawn for European chips 2022," [Online]. Available: https://www2.deloitte.com/uk/en/insights/industry/technology/semiconductor-chip-shortage-supply-chain.html. Accessed on May. 17, 2023.

[3] European Commission, "European Chips Survey," [Online]. Available: https://digital-strategy.ec.europa.eu/en/library/european-chips-survey. Accessed on May. 6, 2023.

[4] K. Vaidyanathan, "Exploiting Challenges of Sub-20 nm CMOS for Affordable Technology Scaling," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, 2014.

[5] R. Puri and D. S. Kung, "The Dawn of 22nm Era: Design and CAD Challenges," in *23rd International Conference on VLSI Design*, Bangalore, India, pp. 429-433, 2010, doi: 10.1109/VLSI.Design.2010.85.

[6] Alka Solanki and Vijendra K. Maurya, "Analysis of Power, Performance and Area at sub-micron ASIC implementation," *Journal of Engineering Sciences*, vol. 11, no. 6, pp. 915-922, 2020.

[7] “Design Challenges in Single-Digit Technology Nodes.", anysilicon.com,[Online]. Available: https://anysilicon.com/design-challenges-in-single-digit-technology-nodes/, Accessed on June. 26, 2023

[8] W. S. Don Bouldin and P. Haug, "ASIC by Design - Automated design of digital signal processing application-specific integrated circuits,"*IEEE Circuits and Devices Magazine*, vol. 20, no. 4, pp. 17-21, 2004.

[9] T. Gaurav, A. Bhatt, and R. Parekh, "Design and Implementation of low power RISC V ISA based coprocessor design for Matrix multiplication," in *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Coimbatore, India, pp. 189-195, 2021, doi: 10.1109/ICESC51422.2021.9532933.

[10] P. Mantovani, R. Margelli, D. Giri, and L. P. Carloni, "HL5: A 32-bit RISC-V Processor Designed with High-Level Synthesis," in *2020 IEEE Custom Integrated Circuits Conference (CICC)*, Boston, USA, pp. 1-8, 2020, doi: 10.1109/CICC48029.2020.9075913.

[11] A. Chang and W. J. Dally, "Explaining the gap between ASIC and custom power: a custom perspective," in *Proceedings 42nd Design Automation Conference*, Anaheim, USA, pp. 281-284, 2005,doi: 10.1145/1065579.1065652.

[12] B. Babu, R. R. Swetha, and K. A. S. Devi, "Comparison of Hierarchical Mixed-Size Placement Algorithms for VLSI Physical Synthesis," in *2011 International Conference on Communication Systems and Network Technologies*,Katra, India, pp. 430-435, 2011, doi: 10.1109/CSNT.2011.95.

[13] N. Ghimire, "Benchmarking of Control Kernels on Open-Source RISC-V Processors," Master's thesis, Aalto University, 2022.

[14] S. K. B. Neha Deshpande, "A Review on ASIC Synthesis Flow Employing Two Industry Standard Tools," *International Journal of Engineering Research and Technology(IJERT)*, vol. 8, no. 17, 2020, doi:10.17577/IJERTCONV8IS17004.

[15] S. Iyengar and L. Shrinivasan, "Power, Performance and Area Optimization of I/O Design," in *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*, Coimbatore, India, 2018, pp. 415-420, doi: 10.1109/ICIRCA.2018.8597347.

[16] H. Geng, T. Chen, Q. Sun, and B. Yu, "Techniques for CAD Tool Parameter Auto-tuning in Physical Synthesis: A Survey (Invited Paper)," in *2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC)*, Taipei, Taiwan, pp. 635-640 ,2022, doi: 10.1109/ASP-DAC52403.2022.9712495.

[17] O. Simola, "Physical implementation of a RISC-V processor," Master's thesis, Aalto University, 2023.

[18]] D. Patterson and A. Waterman, "The RISC-V Reader: An Open Architecture Atlas," Morgan and Claypool Publishers, 2017.

[19] A. Waterman, "The RISC-V Instruction Set Manual. User-Level ISA," 2019.

[20]] A. D. George, "An overview of RISC vs. CISC," in *[1990] Proceedings. The Twenty-Second Southeastern Symposium on System Theory*, Cookeville, USA, pp. 436-438, 1990, doi: 10.1109/SSST.1990.138185.

[21] M. N. Ince, J. Ledet, and M. Gunay, "Building An Open Source Linux Computing System On RISC-V," in *2019 1st International Informatics and Software Engineering Conference (UBMYK)*, Ankara, Turkey, pp. 1-4, 2019, doi: 10.1109/UBMYK48245.2019.8965559.

[22] C. Müllner, "RISC-V Software Ecosystem," [Online]. Available: https://wiki.riscv.org/display/HOME/RISC-V+Software+Ecosystem. Accessed on May 17, 2023.

[23] A. Verma, P. Sharma, and B. P. Das, "RISC-V Core with Approximate Multiplier for Error-Tolerant Applications," in *2022 25th Euromicro Conference on Digital System Design (DSD)*, Maspalomas, Spain, pp. 239-246, 2022, doi: 10.1109/DSD57027.2022.00040.

[24] P. D. Schiavone et al., "Slow and steady wins the race? A comparison of ultra-low-power RISC-V cores for Internet-of-Things applications," in *2017 27th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, Thessaloniki, Greece, pp. 1-8, 2017, doi: 10.1109/PATMOS.2017.8106976.

[25] OpenHW Group, "CV32E40P User Manual," 2022. [Online]. Available: *https://docs.openhwgroup.org/projects/cv32e40p-user-manual/en/latest/*. [Accessed: March 3, 2023].

[26] J. Lienig and J. Scheible, "Methodologies for Physical Design: Models, Styles, Tasks, and Flows," in *Fundamentals of Layout Design for Electronic Circuits*, Springer, Cham, 2020. doi:10.1007/978-3-030-39284-0_4.

[27] K. Ravali, S. Ravi, and H. M. Kittur, "Power Optimization Techniques and Physical Design Flow on Repeaters for High-Speed Processor Core in sub 14nm," in *2018 3rd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology (RTEICT)*, Bangalore, India, pp. 424-428, 2018, doi:10.1109/RTEICT42901.2018.9012129.

[28] H. Bhatnagar, *Advanced ASIC Chip Synthesis*, Springer New York, NY, 2002.

[29] A. Lindqvist, "DEVELOPING LOGIC SYNTHESIS FLOW FOR NVDLA IP,"Master's thesis, Faculty of Information Technology and Communication Sciences, Tampere University, Tampere, Finland, 2022. [Online]. Available:https://trepo.tuni.fi/handle/10024/140133.

[30] B. Canal, "Physical Implementation of a 32-bits RISC microprocessor using XFAB 600nm technology," in *32ª SIMPÓSIO SUL DE MICROELETRÔNICA*, pp. 1-4, 2017.

[31] S. Kister, TMM, Synopsys, Inc., "IC Compiler II: Finding the Best Floorplan, Fast," 2014.

[32] S. Said, "Low Power ASIC Design, a Comparative Study," Master's thesis,California State University, Northridge, 2012.

[33] L. Benini, L. Macchiarulo, A. Macii, and M. Poncino, "From architecture to layout: partitioned memory synthesis for embedded systems-on-chip," in *Proceedings of the 38th Design Automation Conference (IEEE Cat. No.01CH37232)*, Las Vegas, USA, pp. 784-789, 2001, doi:10.1145/378239.379066.

[34] J. G. Prasad, S. R. Karbari, S. Ammikkallingal, and S. K. Bellal, "Analysis, Physical Design and Power Optimization of Design Block at Lower Technology Node," in *2018 3rd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology (RTEICT)*, Bangalore, India, 2018, pp. 732-737, doi:10.1109/RTEICT42901.2018.9012556.

[35] Dhaval S. Shukla, "Floorplan Guidelines for Sub-Micron Technology Node for Networking Chips," [Online]. Available: https://www.design-reuse.com/articles/53962/floorplan-guidelines-for-sub-micron-technology-node-for-networking-chips.html. Accessed on June 29, 2023.

[36] M. Shaikh, B. Soni, and R. Mehta, "Optimization of Floorplan Strategies to Reduce Timing Violation on 28nm ASIC and Scopes of Improvement for Data Center ASICs," in *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, Madurai, India, pp. 93-98,2020, doi:10.1109/ICICCS48265.2020.9121173.

[37] G. Wu and C. Chu, "Detailed Placement Algorithm for VLSI Design With Double-Row Height Standard Cells," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 9, pp. 1569-1573, 2016, doi: 10.1109/TCAD.2015.2511141

[38] Min Pan, "An efficient and effective detailed placement algorithm," in *ICCAD-2005 IEEE/ACM International Conference on Computer-Aided Design*, San Jose, USA, pp. 48-55, 2005, doi: 10.1109/ICCAD.2005.1560039

[39] J. Cong, M. Romesis, J.R. Shinnerl, K. Sze, and M. Xie, "Locality and Utilization in Placement Suboptimality," in Modern Circuit Placement, G.J. Nam and J. Cong (eds.), Boston, MA: Springer, pp. 15-40,2007, doi: 10.1007/978-0-387-68739-1_2.

[40] Yiu-Hing Chan, P. Kudva, L. Lacey, G. Northrop and T. Rosser,"Physical synthesis methodology for high performance microprocessors," *Proceedings 2003 Design Automation Conference (IEEE Cat No03CH37451)*, Anaheim, USA, pp. 696-701, 2003, doi: 10.1109/DAC.2003.1219108.

[41] S. Sreevidya, R. Holla, and R. Raghu, "Low Power Physical Design and Verification in 16nm FinFET Technology," in *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*, Coimbatore, India, pp. 936-940, 2019, doi: 10.1109/ICECA.2019.8822211.

[42] G. Dinesh K, R. Srinidhi, and Srividya. P, "Physical Implementation of Low-Power Area-efficient Digital Delay Locked Loop for High-Speed Interface in Sub -10nm Technology," in *2022 International Conference on Computing, Electronics and Communications Engineering (iCCECE)*, Southend, United Kingdom , pp. 21-24,2022,doi: 10.1109/iCCECE55162.2022.9875101.

[43] G. Rahav, "STA - Static Timing Analysis Electrical Engineering Department," Ben-Gurion University of the Negev, [Online]. Available: https://www.ee.bgu.ac.il/ digivlsi/slides/STA_9_1.pdf. Accessed on May 15, 2023

[44] A. H. Farrahi, M. J. Hathaway, M. Wang, and M. Sarrafzadeh, "Quality of EDA CAD Tools: Definitions, Metrics and Directions," in *Proceedings IEEE 2000*

First International Symposium on Quality Electronic Design (Cat No PR00525), San Jose, USA, pp. 329-336 ,2000, doi: 10.1109/ISQED.2000.838903.

[45] L. Cherif, M. Chentouf, J. Benallal, M. Darmi, R. Elgouri and N. Hmina, "Usage and impact of multi-bit flip-flops low power methodology on physical implementation," in *2018 4th International Conference on Optimization and Applications (ICOA)*, Mohammedia, Morocco, pp. 1-5, 2018 ,doi: 10.1109/ICOA.2018.8370498.

[46] I. Seitanidis, G. Dimitrakopoulos, P.M. Mattheakis, L. Masse-Navette, and D. Chinnery, "Timing-Driven and Placement-Aware Multibit Register Composition," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 8, pp. 1501-1514, 2019, doi: 10.1109/TCAD.2018.2852740.

[47] C. Santos, R. Reis, G. Godoi, M. Barros, and F. Duarte, "Multi-bit flip-flop usage impact on physical synthesis," in *2012 25th Symposium on Integrated Circuits and Systems Design (SBCCI)*, Brasilia, Brazil, 2012, pp. 1-6, doi: 10.1109/SBCCI.2012.6344435.

[48] C. J. Alpert et al., "Design Routability Using Multiplexer Structures," U.S. Patent US 2013/0086537 A1, Apr. 2013.

[49] M. Clarke, M. Rardon, and A. Sood, "Eliminating Routing Congestion Issues with Logic Synthesis," Cadence Design Systems, [Online]. Available: https://www.techonline.com/tech-papers/eliminating-routing-congestion-issues-with-logic-synthesis/. Accessed on May 23, 2023.

[50] J. Lin and Z. Tong, "EDA technology and its implementation in modern electronic technology," in *2011 International Conference on Business Management and Electronic Information*, Guangzhou, China, pp. 816-819, 2011, doi: 10.1109/ICBMEI.2011.5920355.

[51] N. A. N. Hassan, A. B. Abd Manaf, and L. C. Ming, "Optimization of circuitry for power and area efficiency by using combination between latch and register," in *2011 IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE)*, Penang, Malaysia, pp. 240-244, 2011, doi: 10.1109/ICCAIE.2011.6162138.

[52] A. Al-Wattar, S. Areibi, G. Grewal, "An Efficient Framework for Floor-plan Prediction of Dynamic Runtime Reconfigurable Systems," in *International Journal of Reconfigurable and Embedded Systems (IJRES)*,pp. 99-121, 2015, doi:10.11591/ijres.v4.i2.pp99-121.

[53] Preeti Jain, "How Chip Floorplan Design Automation Accelerates Chip Design," [Online]. Available: https://www.synopsys.com/blogs/chip-design/chip-floorplan-design-automation.html. Accessed on July 29, 2023

[54] D.Z.Pan, Minsik Cho, "Synergistic physical synthesis for manufacturability and variability in 45nm designs and beyond," in *2008 Asia and South Pacific Design Automation Conference*, Seoul, Korea (South), pp. 220-225, 2008, doi: 10.1109/ASPDAC.2008.4483945.

A High Level Synthesized Schematic

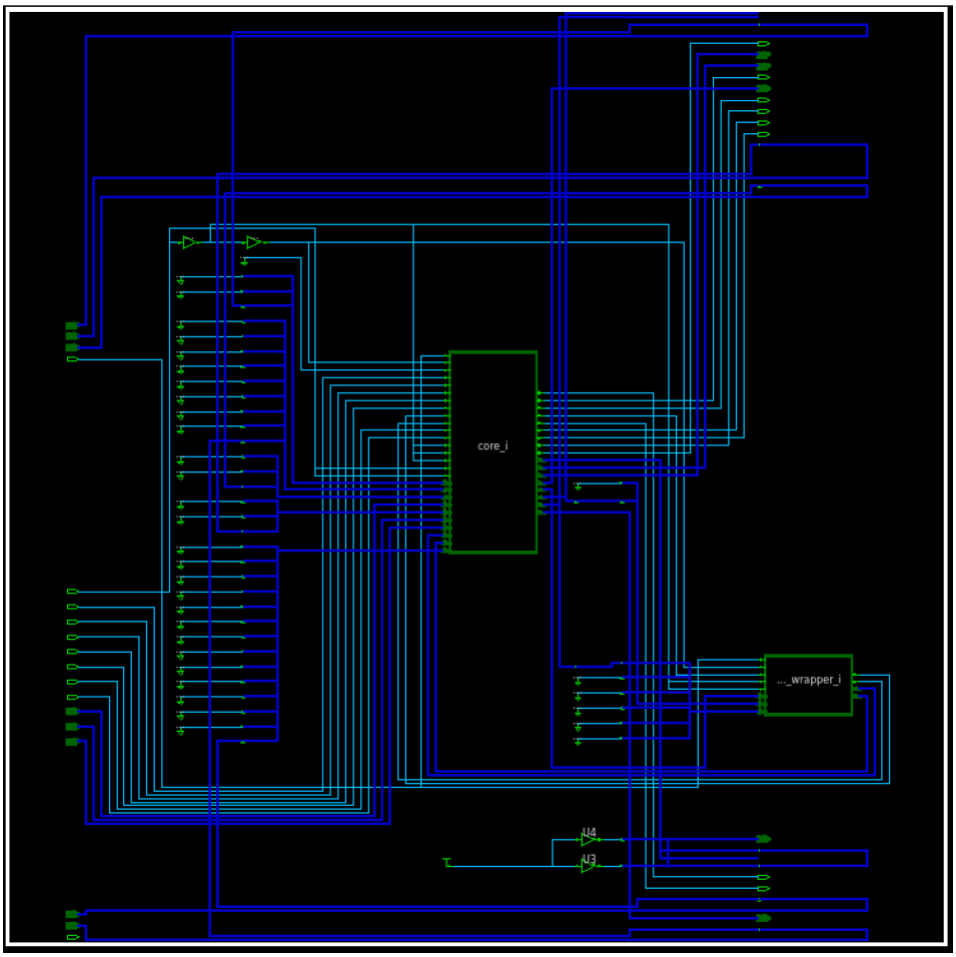


Figure A1: High level synthesized schematic for DS-2

B Floorplan cell density maps

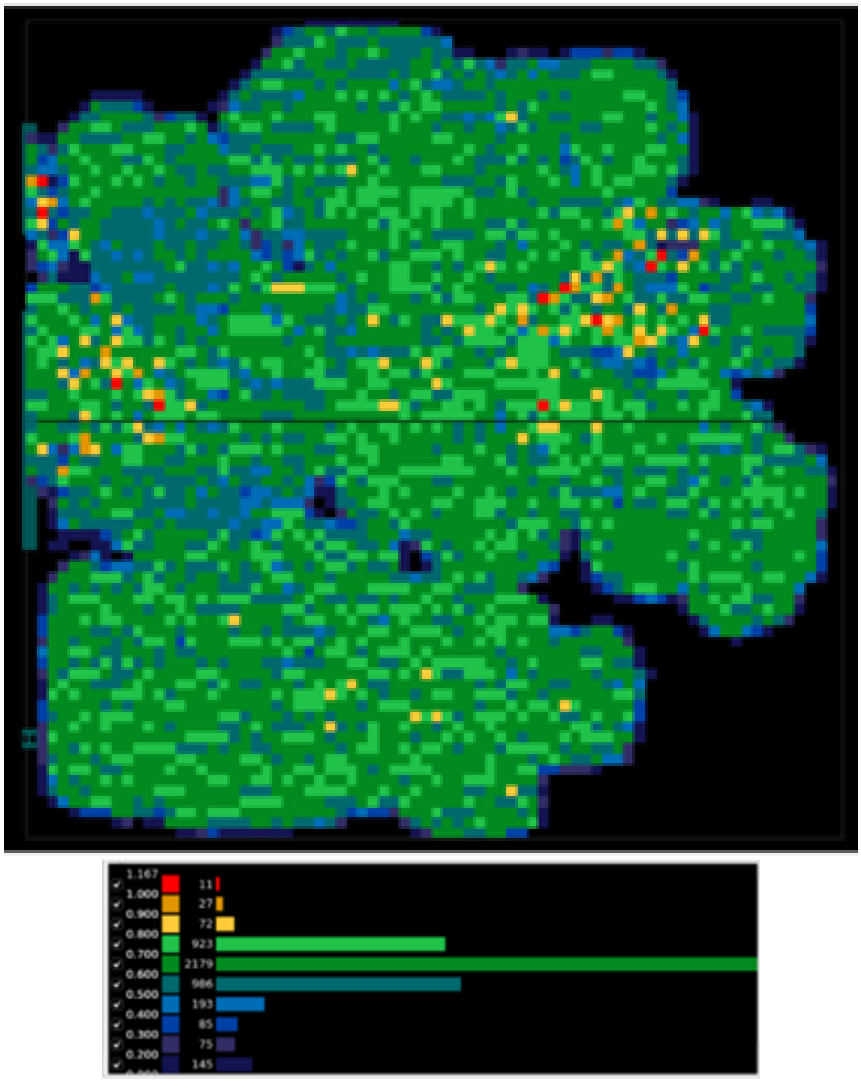


Figure B1: FP-1 Cell density map

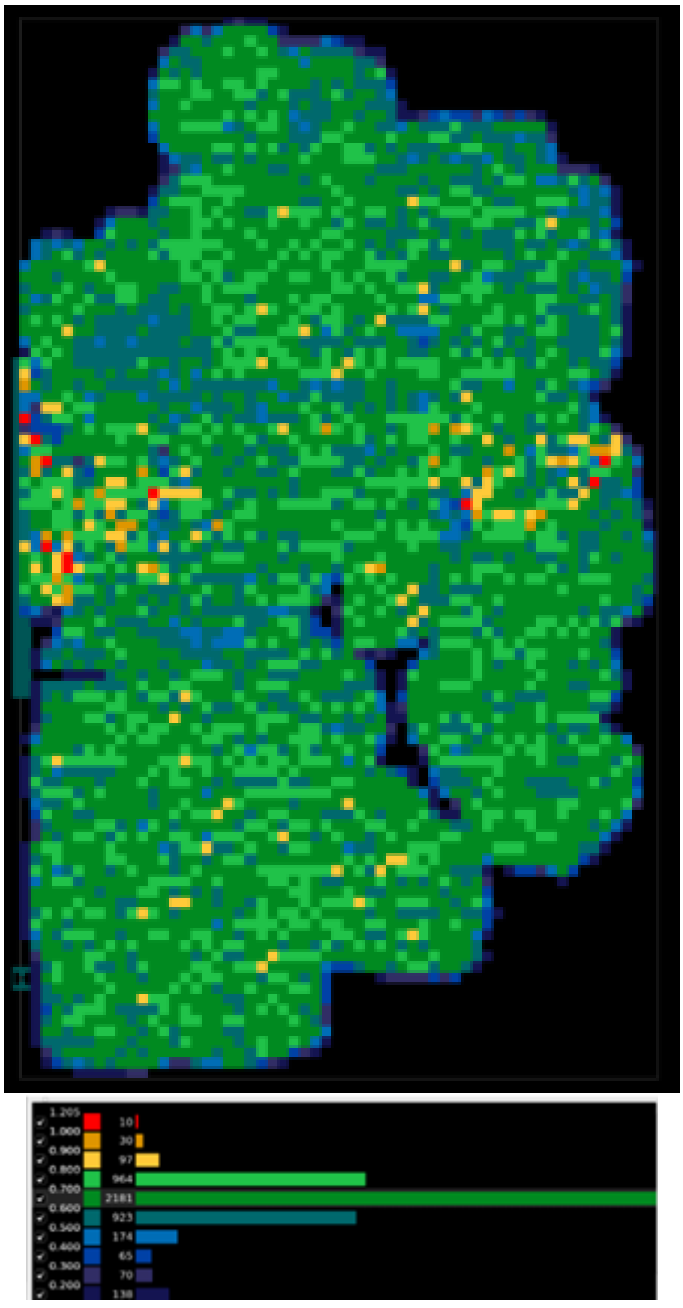


Figure B2: FP-2 Cell density map

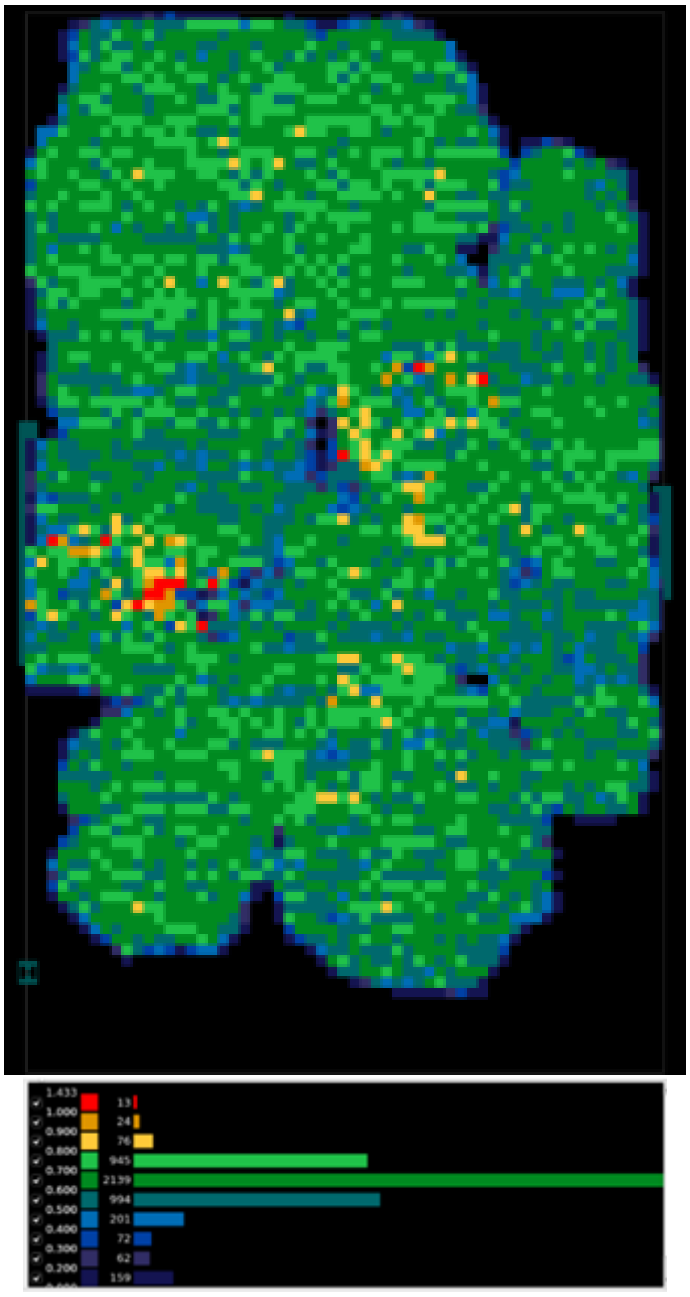


Figure B3: FP-3 Cell density map

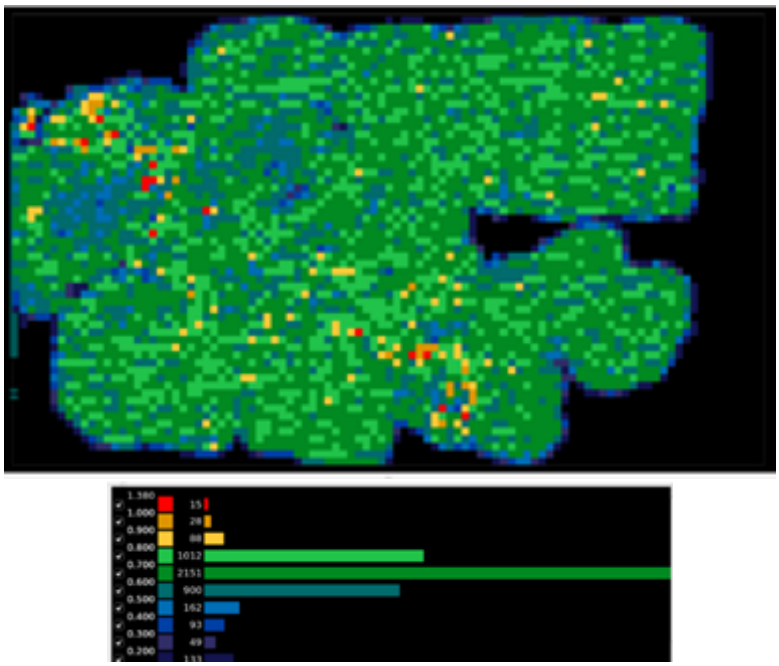


Figure B4: FP-4 Cell density map

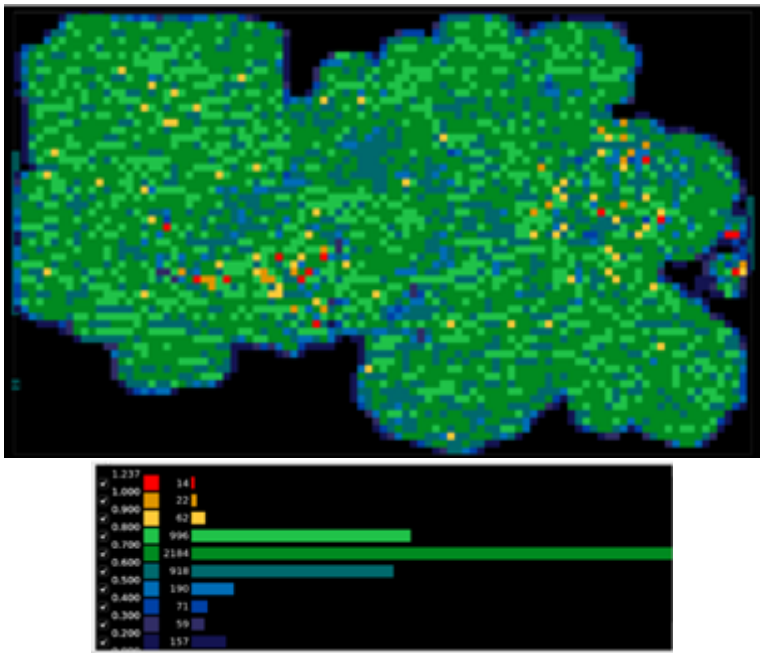


Figure B5: FP-5 Cell density map